WissGrid-Spezifikation: Langzeitarchivierungsdienste

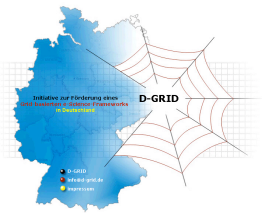
Version - 30. April 2010

Arbeitspaket 3

Verantwortlicher Partner - SUB/DKRZ

WissGrid

Grid für die Wissenschaft



Bundesministerium
für Bildung
und Forschung

Projekt: **WissGrid**

Teil des D-Grid Verbundes und der deutschen e-Science Initiative

BMBF Förderkennzeichen: 01|G09005A-G (Verbundprojekt)

Laufzeit: Mai 2009 - April 2012

Dokumentstatus: stabile Zwischenversion

Verfügbarkeit: öffentlich

Autoren:

Andreas Aschenbrenner, SUB

Harry Enke, AIP

Bernadette Fritzsch, AWI

Michael Lautenschlager, DKRZ

Jens Ludwig, SUB

Jose Mejia, AWI

Torsten Rathmann, DKRZ

Angelika Reiser, TUM

Frank Schluenzen, DESY

Jessica Smejkal, TUM

Revisionsverlauf

Datum	Autor	Kommentare
3.12.2009	AAsche, Jens Ludwig	Erster Entwurf
08.12.2009	Michael Lautenschlager	Ergänzung Einleitung
09.12.2009	Bernadette Fritzsch	Ergänzungen
15.12.2009	Jens Ludwig	Ergänzungen, Zusammenführung, Umstrukturierungen
16.12.2009	Bernadette Fritzsch, Jose Mejia	Spezifikation Dienst Metadatenextraktion
16.12.2009	Michael Lautenschlager	Ergänzung zu Konvertierung und Provenienzdienst
21.12.2009	Jose Mejia	Ergänzung zur Metadatenextraktion, Korrekturen
22.12.2009	Michael Lautenschlager	Überarbeitung nach Telefonkonferenz
07.01.2010	Harry Enke	Korrekturen
10.01.2010	Jens Ludwig	Zusammenführung, Ergänzungen
10.01.2010	AAsche	WDF-Abschnitt
11.01.2010	Jens Ludwig	Ergänzungen, Glossar, Formatierung
30.04.2010	Bernadette Fritzsch, J. Ludwig, Torsten Rathmann, Angelika Reiser, Frank Schluenzen, Jessica Smejkal	Überarbeitungen, Korrekturen

Inhaltsverzeichnis

0 Allgemeines Vorwort zu WissGrid-LZA-Spezifikationen

Diese Spezifikation (zu einem Web/Grid Service, Repositorien, etc.) ist Teil der Gesamtstrategie für Langzeitarchivierung (LZA) in D-Grid, entwickelt durch das Projekt WissGrid. Diese Spezifikation ist eine Komponente der WissGrid-LZA-Architektur und Teil eines wachsenden Pools an Angeboten zur LZA in WissGrid/D-Grid.

Jenseits der Einbettung in die LZA-Aktivitäten des WissGrid-AP3s ist diese Spezifikation darüber hinaus auch geprägt von den Konzepten der anderen WissGrid-APs, von den Entwicklungen von D-Grid Partnern sowie von verwandten Projekten. Alle Abhängigkeiten und Grundannahmen dieser Spezifikation sind in der WissGrid-LZA-Architektur ausführlich beschrieben; einige Punkte daraus sind im Folgenden hervorgehoben:

- Nachnutzung von existierenden Tools: Die Fachdisziplin "LZA" (z.B. Digital Curation Conference, iPRES) ist eine sehr aktive und verzweigte Fachdisziplin, ähnlich wie die Fachdisziplin "Grid" (z.B. Open Grid Forum). LZA im Grid ist nicht nur ein Versuch, die Erfahrungen zweier großer Fachdisziplinen zu verknüpfen. Vielmehr ist für die Nachhaltigkeit der LZA-Dienste und Repositorien in WissGrid ein langfristiger Austausch zwischen diesen Fachdisziplinen essenziell.
- Content Preservation ist der primäre Schwerpunkt (vgl. WissGrid-LZA-Architektur): Die darunter liegende Bit-Preservation wird idealerweise durch die (Grid-)Infrastruktur angeboten; darüber liegende Funktionen der Data Curation (z.B. inhaltliche Selektion der Daten, Beschreibung und wissenschaftliche Verknüpfung) können durch LZA-Dienste und Repositorien unterstützt werden, sind aber primär Aufgaben des jeweiligen Community Grids.
- Modulare Anpassbarkeit an den jeweiligen organisatorischen und technischen Kontext: Auch wenn die WissGrid-LZA-Architektur primär auf Content Preservation ausgerichtet ist, so werden die in AP3 erzeugten Dienste und Repository-Varianten anpassbar an Community-spezifische Anwendungen und Community-spezifische LZA-Strategien sein. In welcher Form die Fachberater und Blaupausen von WissGrid AP2 in Zukunft die Communities in der Entwicklung von LZA-Strategien und Repositorien unterstützen können, wird im weiteren Projektverlauf ausgearbeitet.
- Möglichkeit unterschiedlicher organisatorischer Modelle für den Betrieb: Im Fall von Repositorien könnte man, beispielsweise, auf der einen Seite Softwarepakete

unterscheiden, die von den Communities im Stile einer "Referenzarchitektur" individuell installiert und angepasst werden können; auf der anderen Seite sind aber auch zentral gehostete generische Archive denkbar, die zwar Daten vertrauenswürdig aufnehmen können, aber ggf. nicht für die spezifischen Nutzungsszenarien einer Community optimiert sind.

In diesem Spezifikationsdokument werden nur die technischen Voraussetzungen für unterschiedliche Modelle beschrieben. Die entsprechenden Geschäftsmodelle und Einbettung in D-Grid werden im weiteren Verlauf von WissGrid AP1 ausgearbeitet.

- Ständige Weiterentwicklung der hier beschriebenen technischen Basis: Eine kontinuierliche Weiterentwicklung ist gerade im Bereich der LZA notwendig (die dem schnellen technischen Fortschritt innerhalb nur weniger Jahre voll ausgesetzt ist), aber auch um mit den sich ständig weiter entwickelnden Anwendungen und wissenschaftlichen Methoden der Nutzer mithalten zu können (wo Erfahrungen und Kritik an laufenden Implementierungen die Nutzererwartungen ständig antreiben). In diesem Sinne ist diese Spezifikation nur ein Schnappschuss in der fortlaufenden Evolution von LZA-Diensten bzw. Repositorien.

1 Einführung

Unter Langzeitarchivierung (LZA) wird in WissGrid eine Vielzahl von Aktivitäten zur Sicherung der Nachnutzbarkeit verstanden, wie sie im Anhang 3 des Architekturdokuments erläutert wurden. WissGrid konzentriert sich auf den Teilaspekt der Unterstützung der technischen Nachnutzbarkeit, der sogenannten Content Preservation. Um eine Langzeitarchivierung von Forschungsdaten im Sinne einer Content Preservation im Grid sicherzustellen, ist nicht nur eine langfristig sichere Speicherung notwendig, sondern es gehören auch eine Reihe von weiteren Verarbeitungsschritten dazu. Die Aufgaben im Kontext der Speicherung fallen im weitesten Sinne in den Aufgabenbereich von Repositorien, wie es in der WissGrid-Spezifikation D3.5.1 beschrieben wird. Von den weiteren Verarbeitungsschritten, die zusätzlich zur Speicherung und zum Abrufen von Daten und Metadaten notwendig sind, wurden in der WissGrid-Architektur D3.1 einige identifiziert, die für eine Vielzahl von Grid-Communities sinnvoll sind. Dieses Dokument spezifiziert Dienste, die diese Aufgaben in der D-Grid-Infrastruktur erfüllen können.

Im Einzelnen sind die Dienste, die sich aus der Analyse in der WissGrid-Architektur D3.1 ergaben:

- Repository: Softwaresystem, das Daten mit Metadaten im Grid verwaltet, d.h. unter anderem Speichern, Abrufen und die Veränderung durch Dienste. Es ermöglicht üblicherweise die Operation auf konzeptuellen/intellektuellen Objekten anstatt nur auf der logisch/technischen Dateiebene. Das Repository wird in der parallel entwickelten Spezifikation D3.5.1 "Grid-Repository" behandelt.
- Formatkonvertierung: Erlaubt die Konvertierung von Dateien in ein anderes Format (neue technische Form für dasselbe intellektuelle Objekt). Datenformate müssen z.B. immer dann konvertiert werden, wenn Werkzeuge zum Lesen der Formate nicht mehr zur Verfügung stehen oder nur mit unverhältnismäßig großem Aufwand portiert werden können (z.B. Text und Bildformate). Eine Alternative kann die Konvertierung der Werkzeuge zum Lesen der Formate und Beibehaltung der Originaldatenformate sein, wenn z.B. die Quellen der Werkzeuge zugreifbar und die Datenmengen sehr groß sind. Typisch sind hier Binärformat zum Speichern großer Zahlenmengen aus (beispielsweise) numerischer Modellierung und geologischer Fernerkundung.

- Formatcharakterisierung: Ermöglicht das Auslesen von technischen Metadaten aus Dateien.
- Formatvalidierung: Prüft die Qualität (primär technisch) von Objekten (auf Datei- oder Aggregationsebene) und deren Übereinstimmung mit Definition bzw. Charakterisierung
- Provenienzservice: Erfassen und Dokumentieren von Prozessen, die Daten verändern, z.B. Formatkonvertierung. (Dieser Dienst wird in WissGrid nur als Konzept entwickelt und nicht vollständig umgesetzt werden können.)

Wichtige Anforderungen für die LZA-Dienste sind:

- Nachnutzung und Anpassung von existierenden, etablierten Tools
- Community-spezifische Anpassungen und Erweiterungen sollen möglichst modularisiert sein, z.B. Unterstützung des NetCDF-Formats der Klima-Community als separates Modul für die Formatvalidierung und Metadatenextraktion
- Interoperabilität der Dienste untereinander und mit dem Forschungsarchiv, z.B. Konvertierung als Zugriffsoption von Repositorien und Dokumentation dieser Aktivitäten in einem Provenienzdienst

2 Funktionen in Gesamtarchitektur

Im Architekturdokument wurden die Dienste konzeptionell primär im Bereich der Content Preservation, der technischen Nutzbarkeit und statischen Erhaltung des Inhalts, zugeordnet und zu einem kleineren Teil auch der Data Curation, der Erhaltung der intellektuellen Nutzbarkeit. So trägt ein Dienst zur Metadatenextraktion nicht nur dazu bei, dass die Objekte nach technischen Metadaten weiterhin nutzbar sind, sondern kann auch bei ggf. eingebetteten inhaltlichen Metadaten durch deren Extraktion bei der intellektuellen Nachnutzung helfen. Die Dienste wurden im Architekturdokument aus einer Analyse internationaler Projekte zu Forschungsdateninfrastrukturen gewonnen (wie z.B. ANDS, CASPAR, etc.).

Um die funktionale Rolle zu erläutern, die diese Dienste für die Langzeitarchivierung von Forschungsdaten in der Architektur einnehmen, ist es hilfreich, sie im Arbeitsprozess eines Forschungsdatenarchivs einzuordnen. Forschungsdaten werden in Community spezifischen Arbeitsumgebungen verarbeitet und danach in ein Forschungsdatenarchiv zur

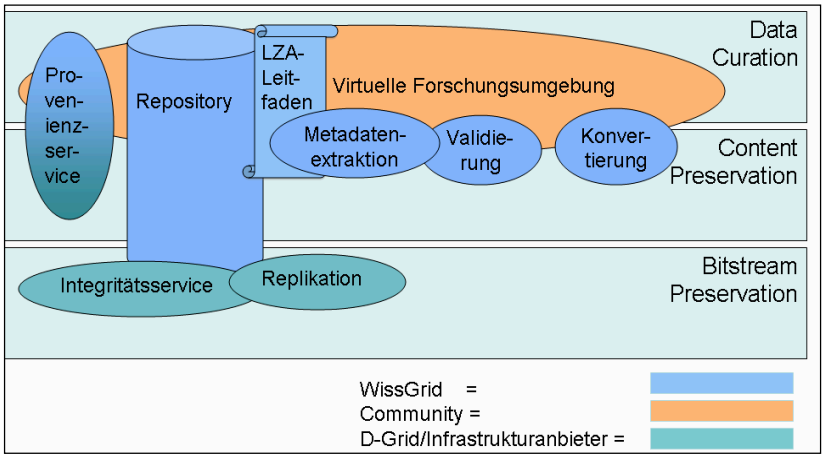


Abbildung 1: Zuordnung von LZA-Diensten zu den LZA-Ebenen und Akteuren.

Langzeitarchivierung übertragen. Im Forschungsdatenarchiv, das durch ein technisches Repository-System realisiert wird, werden die Aufgaben Data Curation, Content Preservation und Bitstream Preservation übernommen und Forschungsdaten so über einen Zeitraum von z.B. 10 Jahren und länger für die Nachnutzung bereitgehalten. Einzelheiten dazu werden in den Deliverables „Generische Langzeitarchivierungsarchitektur im D-Grid“ und „WissGrid-Spezifikation: Grid-Repository“ diskutiert. Insbesondere in der Überführung der Forschungsdaten in die LZA werden eine community-spezifische Data Curation und allgemeine LZA-Dienste relevant. In der allgemeinen Darstellung des OAIS-Referenzmodells, das ein Forschungsdatenarchiv abstrakt beschreibt,

finden sich die hier diskutierten Dienste ungefähr wie in Abbildung 2 im Überblick und in Abbildung 3 für den Ingest gezeigt wieder.

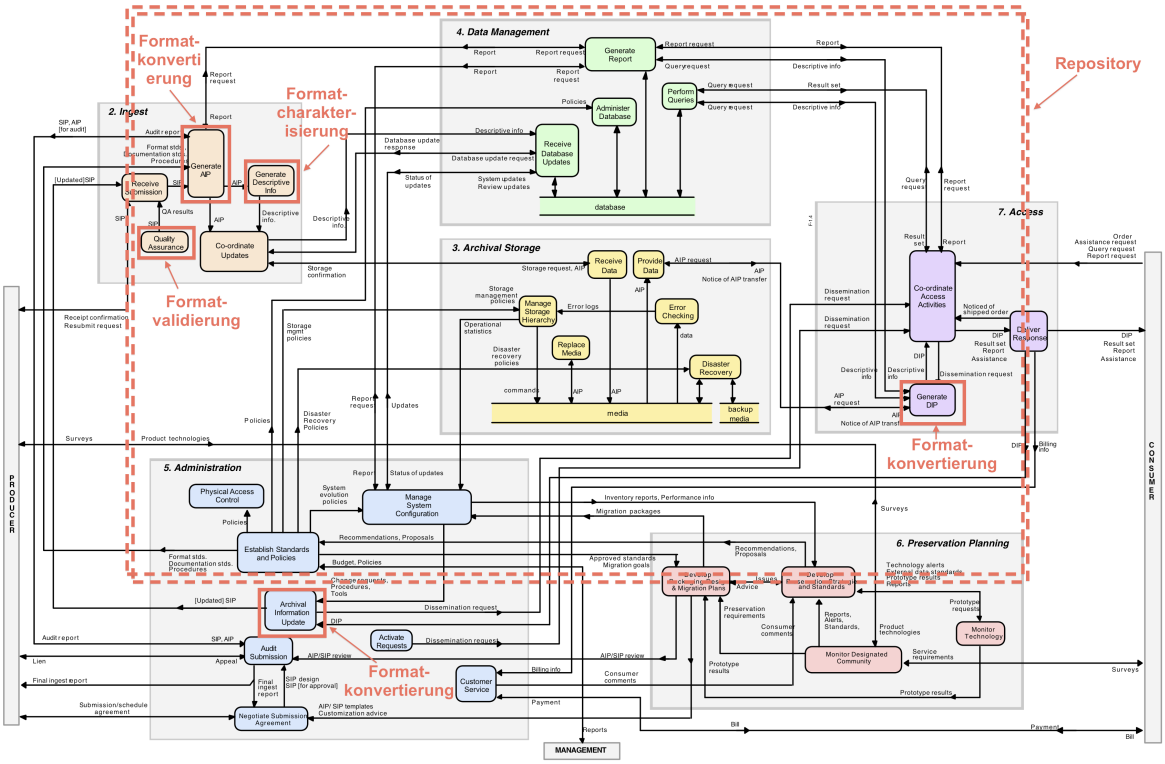


Abbildung 2: WissGrid LZA-Dienste im OAIS-Referenzmodell

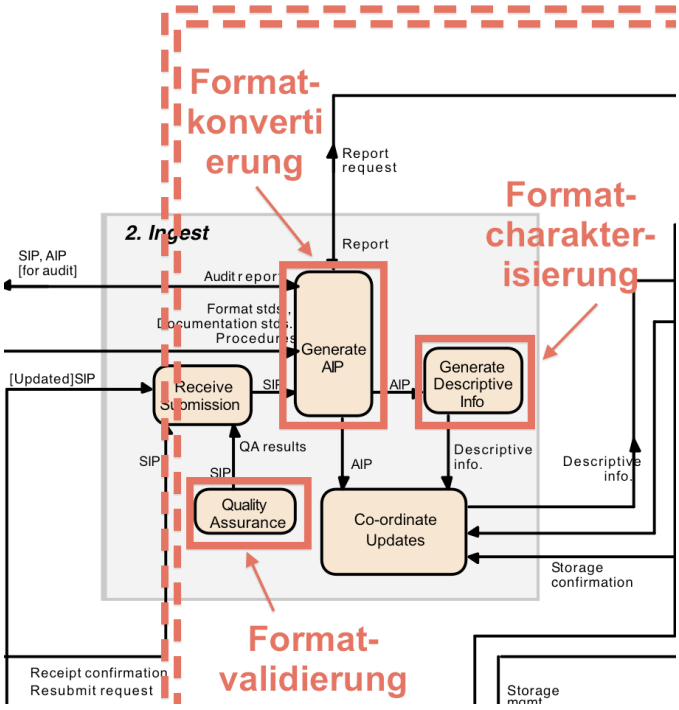


Abbildung 3: WissGrid LZA-Dienste im OAIS-Ingest

Im Einzelnen stellt sich die Integration der Dienste wie folgt dar:

- Formatvalidierung: Die Validierung überprüft, dass das Datenformat technisch fehlerfrei ist, und erfolgt während des Einspielens in das Forschungsdatenarchiv, und kann der "quality assurance" des OAIS zugeordnet werden.[1] Gegebenfalls müssen Fehler, die akzeptabel sind, weil sie die Nutzbarkeit nicht einschränken oder korrigiert werden können, von solchen unterschieden werden, die die Übernahme verhindern.

- Formatcharakterisierung: Um Daten verwalten zu können, sind gewisse Informationen/Metadaten über die Daten notwendig. Während man auf der Forschungsebene dank community-spezifischem Wissen auch mit lückenhaft beschriebenen Daten meist noch sinnvoll arbeiten kann, ist dies in der LZA des Forschungsdatenarchivs nicht mehr möglich. Das bei unvollständiger Datenbeschreibung vorausgesetzte community-spezifische Wissen ist bei interdisziplinärer Datennachnutzung oder bei späteren Forschergenerationen möglicherweise nicht vorhanden.

  Die notwendigen Metadaten müssen zum Teil mitgeliefert werden, weil sie nicht nachträglich erzeugt werden können, zum Teil können diese aber auch durch eine Metadatenextraktion gewonnen werden. Diese wäre im Ingest ein Teil des Prozesses "Generate Descriptive Information". Die Metadaten, die dabei gewonnen werden, können deskriptiv (Titel, Instrument, etc.) und technisch (Format, Auflösung, Kodierungsverfahren, etc.) sein, wenn die Metadaten in der Datei eingebettet (z.B. Header-Informationen) sind. Metadaten, die hingegen nur aus der maschinellen Interpretation der Dateien gewonnen werden, können nur technisch sein.

  Die Ausgabe erfolgt in ein generisches Zwischenformat, das dann je nach Anwendung und Datenmodell noch in ein anwendungsfallspezifisches Metadatenmodell/-format konvertiert (z.B. XML Konvertierung mittels XSLT) und ergänzt werden muss. Für die Arbeit an dem spezifischen Metadatenformaten wird WissGrid höchstens exemplarisch unterstützende Werkzeuge anbieten können.[2]

---

[1] Die Zuordnung zu "Quality assurance" (QA) ist nicht ganz eindeutig, denn QA wird im OAIS als Verifikation des gelungenen Transfers in das Archiv erklärt. Alternativ könnte es auch der Funktionseinheit "Generate AIP" zugeordnet werden, die gewährleistet, dass das Archivpaket den Archivanforderungen entspricht.

[2] Für die Erdsystem-Modellierung werden z.B. Ansätze zur Definition eines standardisierten LZA-Metadatenmodells, der automatischen Erfassung von Metadaten und ihrer Vervollständigung mit Hilfe eines graphischen Werkzeugs im Rahmen des FP7-Projekts METAFOR (http://metaforclimate.eu/) exemplarisch entwickelt.

Eine besondere Aufgabe der Metadatenextraktion ist die Bestimmung des Datenformats. Sie ist unerlässlich für die Entscheidung, ob die angelieferten Forschungsdaten in unterstützten LZA-Datenformaten vorliegen oder ggf. konvertiert werden müssen. Dienste für Formatcharakterisierung und –validierung sowie Formatkonvertierung werden von allen Nutzer-Communities als wichtig klassifiziert und bieten in der strengen Definition von Datenformaten die Chance als Grid-Services implementiert und genutzt werden zu können.

- Formatkonvertierung: Üblicherweise werden Konvertierungsarten nach dem auslösenden Ereignis als "Migration on Ingest", "Migration on Access" und "Migration on Obsolescence" unterschieden. "Migration on Ingest" ist eine Konvertierung, die während des Einspielens z.B. zum Zwecke der Homogenisierung (insbesondere zur Umwandlung der empfangenen Datenformate in unterstützte LZA-Formate) oder für die Erstellung von Archivpaketen (sogenannten AIPs, "Archival Information Packages") aus den eingelieferten Daten (SIP, "Submission Information Packages") durchgeführt wird. Sie würde im OAIS in der Funktionseinheit "Generate AIP" stattfinden. "Migration on Access" ist die Konvertierung der gespeicherten Daten in ein Format, das für den Nutzer (Consumer) sinnvoll ist, und die der OAIS-Funktionseinheit "Generate DIP" (DIP als Zugriffsdatenpaket/"Dissemination Information Package") zugeordnet ist. Die Konvertierung "Migration on Obsolescence" dient der Aktualisierung des Archivinhalts bei Bedarf, unabhängig von der Übernahme neuer Daten oder des Nutzerzugriffs. Sie wird im OAIS der Funktionseinheit "Archival Information Update" im Block "Administration" zugeordnet.
- Provenienz-Service: Um im Nachhinein die Authentizität der Daten bewerten zu können, müssen Informationen über die Herkunft bzw. Quelle der Daten und der erfolgten Maßnahmen und Veränderungen zusammen mit den Daten gespeichert werden,. Diese Provenienzinformationen sind im OAIS in der Definition der Informationspakete enthalten.

Ein Anwendungsfall für die WissGrid-Entwicklungen ist der z.B. Ingest von Modelldaten aus der Erdsystemforschung in ein Forschungsdatenarchiv. Die Modellierung der komplexen Zusammenhänge im Erdsystem liefert i. d. R. hochvolumige Datenströme. Ein weit verbreitetes Datenformat ist dabei NetCDF, das bereits im Header eine Reihe von Informationen über die abgespeicherten Daten und ihre Geo-Referenzierung enthält. Die Langzeitarchivierung der Daten erfolgt in unterschiedlichen Forschungsdatenarchiven. Ein

großer Teil ist dabei im WDC Climate als thematischem Weltdatenzentrum für Modelldaten abgelegt. Das C3Grid bietet den Wissenschaftlern die Möglichkeit, auch auf andere Datenquellen transparent und einheitlich zuzugreifen. Daher werden im Folgenden vor allem das Metadatenmodell am WDCC und das C3Grid Metadatenprofil zugrundegelegt. Häufig werden die Metadaten derzeit noch weitgehend manuell beim Ingest in das Archiv erzeugt. Dies ist zum einen aufwändig und zum anderen fehlerträchtig. Zur Unterstützung der Wissenschaftler werden zwar webbasierte Metadateneditoren bereitgestellt (z.B. geonetwork), die bereits die den Metadatenprofilen zugrundliegenden Standards ISO 19115/139 berücksichtigen. Trotzdem bleibt der Aufwand für den Nutzer noch relativ hoch und die Einarbeitung in die Metadatenproblematik lästig. Zur Vereinfachung sollen Werkzeuge bereitgestellt werden, die diesen Prozess beschleunigen und verbessern. Das in Kapitel 4 beschriebene Tool zur Formatcharakterisierung und seine NetCDF-Erweiterung dient diesem Zweck.

Zusätzlich zu der Sichtweise auf die Langzeitarchivierungsdienste aus Sicht eines Forschungsdatenarchivs kann man die Dienste auch als eigenständige Komponenten im wissenschaftlichen Arbeitsprozess betrachten:

- Formatkonvertierung: Die Umwandlung von Daten von einem Format in ein anderes ist auch unabhängig von der dauerhaften Speicherung ein häufig notwendiger Zwischenschritt, um Daten mit einem anderen technischen Werkzeug weiterzuverarbeiten.
- Formatvalidierung: Eine Prüfung der technischen Korrektheit der Daten ist sinnvoll bei unzuverlässigen Datenquellen oder nach Bearbeitungen, die nicht immer korrekt ablaufen oder Qualitätsverlust bewirken können, wie z.B. Konvertierungen.
- Formatcharakterisierung: Die Extraktion von Metadaten muss nicht erfolgen, um eine Langzeitarchivierung zu unterstützen, sondern sie kann auch für die einmalige oder wiederholte Filterung von Daten nützlich sein. Insbesondere bei der wiederholten Suche nach bestimmten Daten ist eine separate Extraktion und Indizierung der Metadaten sinnvoll, da dass mehrfache Parsen großer Datenmengen sehr ineffizient ist.
- Provenienz-Service: Um die Verarbeitung von Daten (wie z.B. Konvertierungen) nachvollziehen zu können, ist es grundsätzlich sinnvoll, dass ablaufende Prozesse dokumentiert werden. In einer heterogenen und verteilten Umgebung (wie z.B. außerhalb eines Archivs) ist es geboten, dass diese Dokumentation über die Aktionen von Diensten zentral verfügbar und einheitlich ist.

Die technische Einbettung der Dienste in die D-Grid-Infrastruktur wurde auf einer allgemeinen Ebene bereits im Architekturdokument im Kapitel "LZA-Architektur für D-Grid" erläutert. Im Folgenden wird das technische Zusammenspiel der Dienste und ihre Umsetzung im Einzelnen spezifiziert.

3 Integration der Dienste

Um die WissGrid-Langzeitarchivierungsdienste als Grid-Community zu nutzen und in die eigene Umgebung zu integrieren, müssen eine Reihe von Faktoren und Anforderungen berücksichtigt werden. Unterschieden werden müssen dabei insbesondere die Dienste zur Formatkonvertierung, -validierung und -charakterisierung, die auf Dateien bzw. Aggregationen von Dateien operieren, und der Provenienzdienst, der mit Status- und Prozessmeldungen von Diensten operiert.

Für die Formatkonvertierung, -validierung und -charakterisierung:

- Welche Dateiformate sollen als Input verarbeitet werden und welche Datei- und Metadatenformate sind als Ausgabe sinnvoll? Abhängig von den Formatanfordungen müssen die Dienste ggf. um Module erweitert werden.
- Welches Datenvolumen soll verarbeitet werden? Wie groß sind die Dateien und wie viele sind es? [Skalierbarkeit und Verteilbarkeit der Dienste] Welche Verarbeitungsgeschwindigkeit wird benötigt? Diese Faktoren können entscheiden, ob die Dienste als Grid-Jobs oder als Web-Services integriert werden sollten, wenn z.B. ein entsprechender Performanzgewinn durch die Verarbeitung im Grid zu erwarten ist, und ob ein Transfer der Daten zu den Diensten oder umgekehrt sinnvoller ist.
- Wie ist die Verarbeitungskette der Daten im Grid gestaltet? Handelt es sich um einen einmaligen Vorgang oder um eine Routineprozedur? Unabhängig davon, ob es einen Performanzgewinn durch die Grid-Nutzung gibt, kann die Integration als Grid-Job notwendig sein, weil die Verarbeitungskette der Daten im Grid es erfordert.
- Welche Schnittstellen zum Aufruf und zur Übergabe von Ein- und Ausgabedaten sind vorhanden bzw. werden benötigt? [Insbesondere für Dateien. Pull: Links zu Dateien, Push: Dateien selbst bzw. Ausgabe-Device/-Stream?]

Die Umsetzung dieser Dienste kann, wie im Detail im Kapitel 3.2 des Architekturdokuments beschrieben, entweder als Web-Service oder als Grid-Job erfolgen. In beiden Fällen würde eine Kapselung und Erweiterung erfolgen zur Umsetzung der GSI-Fähigkeit und der Bereitstellung einer Service-Beschreibung, z.B. in WSDL. Für Grid-Jobs wird vorausgesetzt, dass die Software der Dienste auf den zu nutzenden Gridressourcen vorinstalliert wird und für

die Job Submission primär das Data Staging behandelt werden muss. Vorgefertigte Skripte können die Job Submission weiter vereinfachen.

Für den Provenienzdienst:

- Wie granular sollen Prozessschritte protokolliert werden? Welche Informationen werden später benötigt, um die Verarbeitung der Daten nachzuvollziehen? Oftmals werden nicht alle Detailinformationen benötigt, sondern je nach Anwendungsfall muss die Community entscheiden, welche Informationen sie übernehmen will.
- In welchem Format können die Status- und Prozessmeldungen vorliegen?
- Wie werden Akteure, Prozesse und Ressourcen nachvollziehbar identifiziert? Während die Identifikation im D-Grid-Rahmen durch die allgemeine Infrastruktur wie AAI und die Middleware gewährleistet ist, müssen für die Community diese Identifikatoren möglicherweise übersetzt und dauerhaft vorgehalten werden.

Ein Provenienzdienst selbst muss eng verknüpft mit der Middleware arbeiten (siehe Kapitel 6), damit aber eine Community einen solchen Dienst nutzen kann, muss sie im wesentlichen nur die Prozessinformationen, die sie benötigt, auswählen und z.B. in ihr Repository übernehmen.

3.1 Integration mit Repositorien

Um LZA-Dienste als Teil von Repositorien (und ggf. auch im Rahmen von komplexeren wissenschaftlichen oder anderen administrativen Abläufen) zu nutzen, ist eine Koordination der Kommunikation der Dienste mit dem Repository notwendig, die durch ein WissGrid LZA-Dienste Framework (WDF) gewährleistet wird. Obwohl das WDF eine zentrale Rolle in der Einbettung von LZA-Diensten in ein Repository hat, ist es eine von dem Repository unabhängige Komponente. Als solches ist das WDF generisch und auch zur Kommunikation mit mehreren, unterschiedlichen Repositorien ausgelegt.

Die Frage, ob die Daten zu den Diensten oder die Dienste zu den Daten transferiert werden sollten, ist bereits für die einfache Nutzung der WissGrid-Dienste relevant. Bei der Integration von externen Diensten in ein Repository kommen aber auch Sicherheitsfragen (dürfen externe/fremde Dienste innerhalb eines vertrauenswürdigen Langzeitarchivs ausgeführt werden?) und letztlich die Architektur des Repositories (lässt die Softwarearchitektur und die (verteilte?) Hardware die Ausführung von LZA-Diensten zu?) als Faktoren hinzu. Im Rahmen des WissGrid-Projektes wird aufgrund von Sicherheitsbedenken, und mit dem Ziel einer

möglichst generischen Lösung nur das Muster "Daten zu den Diensten" verfolgt. Partnerprojekte (z.B. AstroGrid) experimentieren aber auch mit Community-spezifischen Ansätzen für "Dienste zu den Daten", und die dort gesammelten Erfahrungen werden in WissGrid einfließen.

Die Funktionalität und Architektur des WDF wird im Kapitel 7 beschrieben. Eine Beschreibung der Schnittstelle zu den LZA-Diensten findet sich in als Teil der Spezifikation des jeweiligen Dienstes im Verlauf dieses Dokuments. Die Schnittstelle zu den Repositorien ist in Kapitel 2 der Repositorien-Spezifikation (paralleles Dokument) beschrieben.

4 Formatcharakterisierung und -validierung

Beide Dienste und ihre Erweiterung werden zusammen behandelt, da sie verwandte Funktionalitäten bereitstellen und das vorgeschlagene Umsetzungsframework beide unterstützt.

4.1 Funktionalität

Der funktionale Ablauf stellt sich wie folgt dar:

1. Identifikation des Datenformats
2. Identifikation des richtigen Formatmoduls
3. Formatmodul parst Daten
4. Anwendung der Validierungskriterien
5. Serialisierung der gewonnenen Metadaten in Zielformat

Als primäre Eingabeparameter des Dienstes sind im WSDL oder als Teil der Job Submission zu übergeben:

- Eingabedaten (einzelne Dateien oder Verzeichnisse) als URIs
- ggf. Validierungskriterien (falls zusätzliche Kriterien neben den sich aus der Datenformatspezifikation ergebenden, angewandt werden sollen)
- ggf. Ausgabedatei für Metadaten und/oder Konformitätsgrad (falls diese nicht als Teil eines normalen Rückgabewert geliefert werden sollen)

4.2 Technische Umsetzung

International gibt es bereits eine Reihe von Aktivitäten, von denen das JHOVE2 (https://confluence.ucop.edu/display/JHOVE2Info/Home) Framework derzeit am erfolgversprechendsten zu sein scheint, da es sehr modular aufgebaut ist und sowohl die Identifikation von Dateiformaten, die Extraktion von Metadaten als auch die Validierung ermöglicht. Für einzelne dieser Aufgaben gibt es eine Reihe von Alternativen, wie z.B. DROID (http://sourceforge.net/projects/droid/, nur Identifikation von Dateiformaten und eingebettet in JHOVE2), das Metadata Extraction Tool der National Library of New Zealand

(http://meta-extractor.sourceforge.net/, nur Extraktion von Metadaten), EXIFTool (http://www.sno.phy.queensu.ca/~phil/exiftool/, Extraktion und Modifikation von Metadaten aus Bild-, Audio- und Video-Daten) oder FITS (http://code.google.com/p/fits/, integriert eine Reihe der genannten Werkzeuge) sowie formatspezifische Tools wie z.B. Adobe Preflight für PDFs, die auch die Validierung übernehmen.

JHOVE2 basiert auf dem bereits in der Bibliothekswelt verbreiteten JHOVE[3] und wurde überarbeitet und erweitert, um unter anderem auch die Analyse komplexer digitaler Objekte zu unterstützen. JHOVE2 unterstützt die Datenformate ICC Color Profile, JPEG2000, PDF, SGML, Shapefile, TIFF, UTF-8, WAVE und XML. Es bietet eine Reihe von Optionen, die als Eingabeparameter eines Web Services, einer Job Submission oder als Vorkonfiguration berücksichtigt werden sollten und sich in der Kommandozeilensyntax wie folgt darstellen:

- "`% jhove2 [-ik] [-b size] [-B Direct|NonDirect|Mapped] [-d JSON|Text|XML] [-f limit] [-o file] name ...`
- -i specifies that the unique identifiers for each reportable property are shown;
- -k specifies that message digests are to be calculated;
- -b size specifies the I/O buffer size (defaults to 131072 bytes);
- -B specifies the buffer type: `Direct` (default), `NonDirect`, or `Mapped`;
- -d specifies the display form: `JSON`, `Text` (default), or `XML`;
- -f limit specifies the fail fast limit (defaults to `0`);
- -o limit specifies the name of an output (defaults to the standard output unit); and
- name is the file or directory name or URI to be characterized."[4]

Dabei sind die "unique identifiers for each reportable property" Identifier/Namen für die formatspezifischen Metadaten, die ausgegeben werden. Als "message digests" werden Checksummen, wie z.B. CRC32, MD5, SHA-1 und weitere, in den Metadaten ausgegeben. Durch den modularen Aufbau kann JHOVE relativ einfach für andere Anwendungsklassen erweitert werden, so dass auch weitere community-spezifische Formate unterstützt werden können, wie im folgenden Abschnitt exemplarisch gezeigt wird. WissGrid kann in diesem Kontext für neue Communities Unterstützung leisten, indem es einerseits neue Module bereitstellt, die von anderen Disziplinen nachgenutzt werden können, und andererseits mit der

[3] http://hul.harvard.edu/jhove/index.html
[4] https://confluence.ucop.edu/download/attachments/9470832/JHOVE2-Architecture-v11.pdf?version=1

exemplarischen Implementierung Wege aufzeigt, wie weitere Module entwickelt und eingebunden werden können.

4.3 Entwicklungsaufgaben

Dienst:

- untersuchen welche Formatmodule wichtig/relevant für viele Communities sind und wo sich eine Modulentwicklung lohnt
- Entwicklung entsprechender Module
- SLA-Beschreibung erstellen

GridJob:

- Anpassungen des Dienstes, so dass er als Grid-Job lauffähig ist
- Installation
- Skript zur Ausführung des Dienstes erstellen

Web Service:

- formalisierte Dienstebeschreibung erstellen (WSDL)

Interaktion mit WDF

- Analyse: Wie lässt sich der funktionale Ablauf mit dem WDF abbilden?
- Analyse: Unter welchen Bedingungen bzw in welchen Situationen lohnen welche Aspekte der Grid-Nutzung (z.B. Verteilung/Parallelisierung)
- Implementierung des funktionalen Ablaufs im WDF

- Testen
- Anleitung schreiben
- Installationspaket erstellen

4.4 Anwendungsfälle

4.4.1 Exemplarische technische Umsetzung in der Klima-Community

Exemplarisch wurde für die Community der Klimaforschung bzw. der Klimafolgenforschung ein Modul für NetCDF-Files (J-NetCDF Metadata Extractor, Kurzform JaNEME) geschrieben, das in das JHOVE Framework eingebunden ist. Damit können technische Metadaten (u.a. Validierung der NetCDF-Version) sowie inhaltliche Metadaten aus den Headern extrahiert und in ein vorhandenes Metadatenmodell eines LZA-Repositorys eingetragen werden, so dass der manuelle Aufwand für die Vervollständigung der Metadaten

minimiert werden kann. Das Modul ist dabei soweit wie möglich generisch konzipiert, um eine einfache Übertragung auf andere Forschungsdatenarchive zu erlauben.

Die technologische Basis für die Entwicklung bildet das Spring Framework. Damit werden die notwendigen Java Entwicklungen entscheidend vereinfacht, was für eine Übertragung der hier skizzierten Arbeiten enorm wichtig ist. Außerdem werden dadurch Anwendungen mit unterschiedlichen Application Servern (Tomcat, Jetty, …) und sogar ohne Application Server (als Applet oder standalone Swing Anwendung) möglich. Die Stärke der Verwendung des Spring Frameworks besteht in der sogenannten Inversion-of-Control, einem Mechanismus für die einfache Definition der Komponenten und ihrer Abhängigkeiten während ihres gesamten Lebenszyklus. So wird eine Komponente wie ein Formatmodul, Displayer oder eine Datenbank-JDBC-Verbindung in dem Framework registriert und durch ihre API zu einem späteren Zeitpunkt aufgerufen.

Als ein Beispiel für den generischen Charakter des JaNEME Moduls seien hier die Variablenlisten genannt, die konzeptionell gleich als erweiterbar und austauschbar vorgesehen sind. Im Rahmen des C3-Metadatenprofils[5] sind die Variablennamen vorzugsweise aus der NetCDF Climate and Forecast (CF) Metadata Convention[6] zu nehmen, die die gebräuchlichsten Variablen einschließlich der kanonischen Einheiten aufführt. Das Modul prüft nun alle im Header des NetCDF-Files aufgeführten Variablen danach, ob sie in der Konvention enthalten sind und ihre Einheiten mit der angegebenen übereinstimmen. Falls das nicht der Fall sein sollte, erhält der Nutzer eine Warnung, so dass nochmals überprüft werden kann, ob tatsächlich ein neuer Variablenname benutzt werden soll oder ob Schreibfehler vorliegen. Dabei kann die CF-Liste auch durch andere Listen ersetzt oder ergänzt werden, je nachdem welche Metadatenmodelle in dem jeweiligen Forschungsarchiv verwendet werden.

JaNEME ist auch als Modell für die Entwicklung anderer Formatmodule hilfreich. Vor allem sind seine Validatoren nicht auf NetCDF-Attribute eingeschränkt, sondern sie können für beliebige Fileheader eingesetzt werden. Die Dokumentation von JaNEME befindet sich im Anhangkapitel 8.1.

[5] http://www.c3grid.de/fileadmin/c3outreach/generation-1/metadata_profile.pdf
[6] http://cf-pcmdi.llnl.gov/documents/cf-standard-names/

5 Formatkonvertierung

5.1 Funktionalität

In ihrer Grundfunktion leistet eine Formatkonvertierung die Umwandlung eines digitalen Objektes von einem Format in ein anderes. In unserem Fall ist das zu konvertierende Objekt eine Datei oder Teil einer Datei. Die Grundfunktion lässt sich in den folgenden funktionalen Ablauf einbetten.

1. ggf. Identifikation des Datenformats (z.B. mit JHOVE)
2. Identifikation des richtigen Konvertermoduls, ggf. im Zusammenspiel mit einer Serviceregistry
3. Ausführung des Konvertermoduls
4. ggf. Validierung des Zielformats
5. ggf. Generierung von Provenienzmetadaten[7]

Als primäre Eingabeparameter des Dienstes sind zu übergeben:

- Eingabedaten (Datei) als URIs
- Zielformat und ggf. weitere Parameter[8]
- ggf. URI der Ausgabedatei oder des Ausgabeverzeichnisses (falls der Dienst diese nicht als Rückgabewert selbst liefern soll)
- ggf. explizite Angabe des zu benutzenden Konverters

Dieser Ablauf ist nur in der Lage die Kernfunktionalität abzudecken. Formatkonvertierungen können aber beliebige Komplexität besitzen. So ist es manchmal erforderlich, Beziehungen zwischen Dateien, z.B. Weblinks, zu erhalten. Oder die konvertierenden Dateien müssen erst nach bestimmten, vorgegebenen Regeln zusammengestellt werden, wie z.B. in der Aufgabe „Konvertiere alle TIFF-Daten, die älter als 2 Jahre sind, in ein JPEG 2000 Format“.

[7] Hinzufügung von Detailinformation über die Formatkonvertierung zu den Metadaten
[8] Beispielsweise Auswahlparameter, wenn nur ein Teil einer Datei konvertiert werden soll, oder Qualitäts- und andere Formatparameter für Bilddateien

5.2 Technische Umsetzung

Die Kernfunktionalität kann wie in Kapitel 3.2 des Architekturdokuments beschrieben als Web-Service oder als Grid-Job zur Verfügung gestellt werden. Im Falle des Grid-Jobs wird der oben angeführte Ablauf in einen geeigneten Grid-Workflow eingebettet, vorzugsweise in den im Abschnitt 7.1 vorgestellten WissGrid-Dienste-Ablauf. Die primären Eingabeparameter für die Konvertierung werden im WSDL oder als Teil der Job-Submission übergeben.

Für die Realisierung erweiterter Funktionalität werden zusätzliche Komponenten benötigt. An Nicht-Grid-Software zur Zusammenstellung von zu konvertierenden Dateien nach vorgegebenen Regeln wird bereits gearbeitet. Die Entwicklung der beiden folgenden Produkte ist weit fortgeschritten.

- CriB, http://crib.dsi.uminho.pt: CriB wurde von der University of Minho, Portugal, entwickelt, um Institutionen zur Bewahrung des Kulturerbes bei der Datenmigration und –erhaltung zu unterstützen. CriB ist ein Service Oriented Architecture (SOA) Framework, das zur eigentlichen Formatkonvertierung auf externe Dienste zurückgreift. Auf der Webseite wird ein Prototyp vorgestellt. Die nachhaltige Unterstützung von CriB ist nicht sicher.
- kopal MigrationManager: Der kopal MigrationManager ist ein Prototyp innerhalb der koLibRI-Software[9], der im Zusammenspiel mit dem kopal-Langzeitarchiv Objekte mit Migrationsbedarf anhand granularer Metadaten identifiziert und dann zur Konvertierung übergibt. Es werden externe Konvertierungswerkzeuge benutzt.

Falls Bedarf an solcher, erweiterter Funktionalität besteht, könnte versucht werden, diese als Web-Service zu realisieren und dabei eines der beiden genannten Produkte direkt einzubinden. Bei einer Realisierung als Grid-Job wird eine direkte Verwendung von CriB oder kopal voraussichtlich nicht möglich sein. Hier könnte eine Eigenentwicklung geschaffen werden, die auf der Technik von CriB oder kopal beruht und in Schritt 2 des WissGrid-Dienste-Ablaufs (siehe Abschnitt 7.1) integriert wird.

5.3 Entwicklungsaufgaben

Dienst:

- untersuchen welche Formatmodule wichtig/relevant für viele Communities sind und wo sich eine Modulentwicklung lohnt

[9] http://kopal.langzeitarchivierung.de/index_koLibRI.php.de

- Entwicklung entsprechender Module
- SLA-Beschreibung erstellen

GridJob:

- Anpassungen des Dienstes, so dass er als Grid-Job lauffähig ist
- Installation
- Skript zur Ausführung des Dienstes erstellen

Web Service:

- formalisierte Dienstebeschreibung erstellen (WSDL)

Interaktion mit WDF

- Analyse: Wie lässt sich der funktionale Ablauf mit dem WDF abbilden?
- Analyse: Unter welchen Bedingungen bzw in welchen Situationen lohnen welche Aspekte der Grid-Nutzung (z.B. Verteilung/Parallelisierung)
- Implementierung des funktionalen Ablaufs im WDF

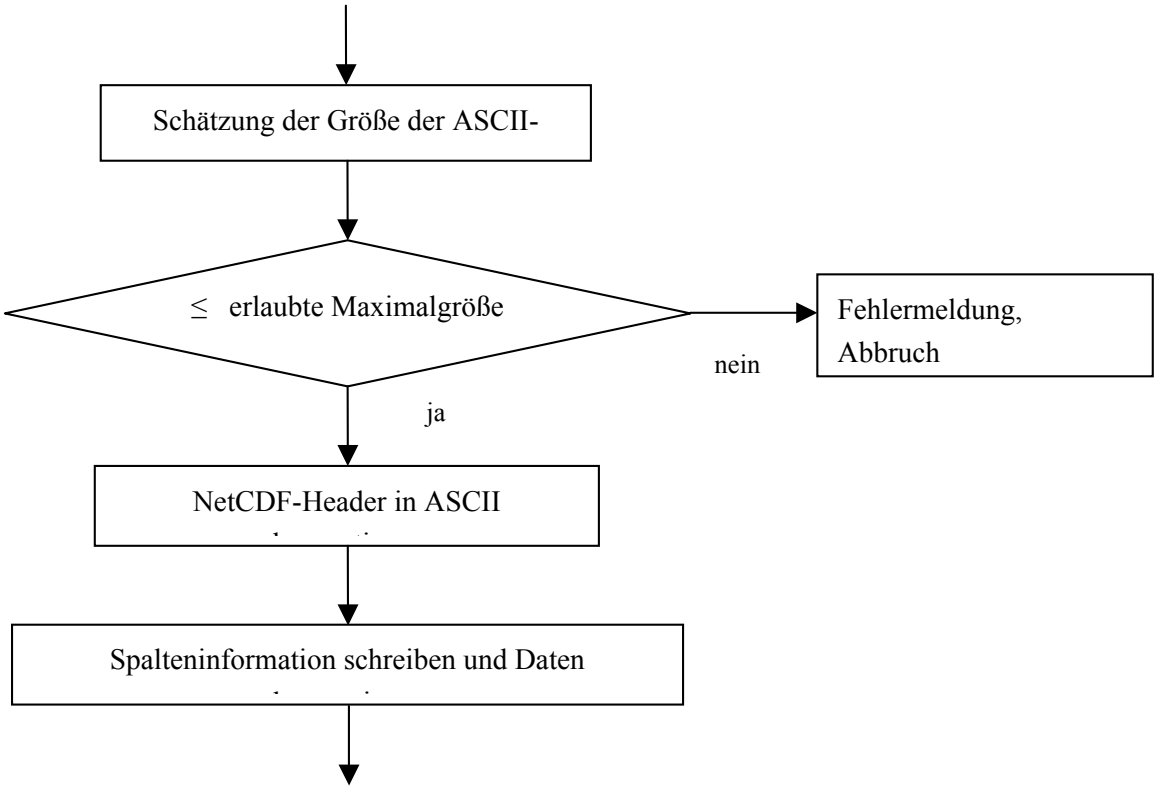
- Testen
- Anleitung schreiben
- Installationspaket erstellen

5.4 Anwendungsfälle

5.4.1 Technische Umsetzung in der Klimafolgenforschung

Seitens der Klimafolgenforschung wird ein Werkzeug zur Formatkonvertierung von dem in der Klima-Community genutzten Binärformat NetCDF nach ASCII CSV gewünscht, um die Daten zur Nachnutzung in Excel oder Geodaten-Informationssysteme (GIS) einladen zu können. Die NetCDF→ASCII-Konvertierung sollte im C3-Grid verwirklicht werden, weil dort die Nutzer sind und im C3-Grid sowohl eine Grid-Infrastruktur als auch Langzeitarchive mit NetCDF-Dateien bereits vorhanden sind.

Der neue Dienst sollte zunächst als Web-Service zur Conversion on Access realisiert werden, später dann als Grid-Job. Schritt 3 in Abschnitt 5.1 sollte den folgenden Ablauf besitzen:



Im C3-Grid ist die Größe einer Download-Datei auf zwei Gigabyte begrenzt. Diese Grenze ist gesetzt worden, weil viele Nutzer mit 32-Bit-Systemen unter Windows arbeiten und ein einzelner 32-Bit-Prozess unter Windows ohne zusätzliche Einstellungen nicht mehr als 2 GB belegen kann. Außerdem soll die Ausgabedatei nicht über eine Größe wachsen, die einen zügigen Download erschwert. Da die Größe von NetCDF-Dateien im Terabyte-Bereich liegen und die ASCII-Datei noch einmal um den Faktor 5-8 größer sein kann, soll die Größe der ASCII-Datei vor Beginn der eigentlichen Konvertierung geschätzt werden. Falls die Schätzung mehr als 2 GB ergibt, soll der Ablauf mit einer Fehlermeldung abgebrochen werden.

Für normale Nutzer ist der Zugang zum C3-Grid das C3-Portal. Über das C3-Portal kann der Nutzer schon jetzt nicht nur nach Daten suchen, sondern die Ausgabe auf bestimmte Variablen, z.B. die Temperatur, einschränken und Grenzen für die auszugebenden geographischen Längen und Breiten, die Höhen und Zeiten setzen. Auf diese Weise kann die Überschreitung der maximalen Download-Größe vermieden werden. Die vom Nutzer im C3-Portal gesetzten Auswahlparameter sollen an die Größenschätzung und an den Formatkonverter übergeben werden.

Wenn die Größenschätzung keine Überschreitung ergibt, soll im nächsten Schritt ein Dump des NetCDF-Headers durchgeführt werden. Hinzu kommen Informationen, welche Größen

sich wo im nachfolgenden numerischen Output befinden, zusammen mit den vom Nutzer gesetzten Auswahlparametern. Nach dieser Header-Information sollen die numerischen Daten gemäß den Vorgaben des Nutzers ausgegeben werden. Alle wesentlichen Informationen zur Art und Herkunft der Daten werden also in den Kopf der ASCII-Datei übernommen. Das Einladen in die Applikation des Nutzers sollte dies nicht stören. Excel kann beispielsweise angewiesen werden, erst von einer bestimmten Zeile an Daten einzulesen.

Für die Größenschätzung von NetCDF- und GRIB-Download-Dateien gibt es im C3-Grid ein Python-Skript, das so erweitert werden könnte, dass auch die Größe von ASCII-Dateien geschätzt werden kann. Für den Header-Dump und die Formatkonvertierung stehen am DKRZ mehrere Programme zu Verfügung, deren Eignung für das Grid geprüft werden soll. Dies sind CDOs (Climate Data Operators), NCOs (NetCDF Operators) und ein noch nicht freigegebenes Java-Tool.

5.4.2 Datenformate in den Photon Sciences

Photon Science (PS) Communities sind außerordentlich multidisziplinär und heterogen. Dies spiegelt sich in den verwendeten Datenformaten wieder, die das ganze Spektrum von ASCII bis VRML abdecken. Selbst in einzelnen Disziplinen wie der Protein-Kristallographie (MX) oder Small-Angle-Xray-Scattering (SAXS) werden eine Vielzahl von sehr unterschiedlichen Formaten eingesetzt, was sowohl die Langzeitarchivierung als auch den Austausch von Daten zwischen verschiedenen Anwendungen und Disziplinen erheblich erschwert.

Die PS Communities sind sich dieses Missstandes durchaus bewusst. In einem Artikel von M.T. Dougherty et al.[10] steht zu lesen: „*The biosciences need an image format capable of high performance and long-term maintenance.*“. Als gemeinsames, standardisiertes Datenformat wird hier insbesondere HDF5 vorgeschlagen.

Im Rahmen der PNI (Photonen, Neutronen, Ionen) Initiative der Helmholtz-Gesellschaft wird im Rahmen eines Antrags (High-Data-Rate-Initiative, HDRI) dieses Problem aufgegriffen und ebenfalls HDF5 als Standard propagiert.

Das PaN-Data Projekt (http://www.pan-data.eu), ein Zusammenschluss europäischer Synchrotron- und Neutronen-Quellen, bemüht sich verbindliche Daten-Policies und Standards zu entwickeln. In diesem FP7-Projekt wird NeXus (http://www.nexusformat.org), eine HDF5-

[10] M.T. Dougherty et al., Communications of the ACM, 52(10):42-47, 2009, *Unifying Biological Image Formats with HDF5*

Erweiterung mit einem hierarchischen XML-Metadaten-Layer, als Standard-Format vorgeschlagen und entsprechende Entwicklungen vorangetrieben.

Verschiedene Experimente an Free Electron Laser-Facilities wie LCLS (http://lcls.slac.stanford.edu) oder FLASH (http://flash.desy.de/) setzen bereits auf HDF5, da dies Format den Anforderungen an Stabilität, Datendurchsatz und Langzeitarchivierung am ehesten gerecht zu werden scheint. Zudem hat HDF5 den Vorteil, von kommerziellen Anwendungen wie Matlab und IDL direkt unterstützt zu werden,

Es scheint zumindest sehr wahrscheinlich, dass sich HDF5 oder HDF5-basierte Datenformate als Standard in den PS Communities etablieren werden. Dies wird mit Sicherheit die Entwicklung und Implementierung von Konvertierungsdiensten erforderlich machen. Das lässt sich am Beispiel der Protein-Kristallographie illustrieren.

Daten	**Formate**
Images (Experiment-Daten)	Modifiziertes TIF, imageCIF, proprietär
Strukturfaktoren (Intensitäten, Phasen)	MTZ, ASCII, CIF, XDS
Protein-Koordinaten	PDB, CIF
Protein-Sequenzen	Fasta
Chemische Daten	IUPAC
Metadaten (Methoden, Analysis-Pathway, etc)	PDB, CIF, ASCII

Es existieren bereits Konvertierungs-Programme, um zum Beispiel zwischen CIF und PDB oder MTZ und CIF zu konvertieren, aber kaum Algorithmen für die Umwandlung oder Einbettung in HDF5. Im Falle der Röntgen-Tomographie oder Single-Particle-Imaging gibt es noch einige Probleme zu überwinden, da hier ein Datensatz aus vielen tausend Einzelaufnahmen besteht. Die resultierenden HDF5-Dateien hätten eine Größe im Terabyte-Bereich, was die Handhabung, insbesondere die Archivierung, erschwert.

5.4.3 Datenformate in den Sozialwissenschaften

In den Sozialwissenschaften liegt das Problem weniger in den Formaten der statistischen Rohdaten selbst, sondern in den Syntax-Dateien, mit denen die statistischen Daten analysiert

werden. In Rahmen des SOEB (Berichterstattung zur sozioökonomischen Entwicklung Deutschlands, http://www.soeb.de) werden überwiegend SPSS und STATA zur Analyse eingesetzt, SAS, Systat oder R spielen bislang nur eine untergeordnete Rolle. Für die Analyse in einer verteilten Umgebung wird allerdings aus Kostengründen R favorisiert. Jedes dieser Statistik-Programme verwendet eine andere, inkompatible Syntax-Sprache.

Die Syntax ist darüber hinaus häufig abhängig von der Version der verwendeten Statistik-Software. So wird dieselbe Syntax mit SPSS-9 ein anderes Ergebnis ergeben als mit SPSS-13. STATA-Formate wechseln mit praktisch jedem neuem Release. Zwar gibt es eine Rückwärts-Kompatibilität und die Möglichkeit Dateien in Formaten älterer Versionen abzuspeichern sowie verschiedene Formate von Excel bis SAS Export-Dateien zu importieren, doch sind diese Optionen nicht für eine Grid-Umgebung geeignet und im Sinne der Reproduzierbarkeit der Analysen wenig transparent.

Um die Nachhaltigkeit der Analysen zu gewährleisten, sehen die Sozialwissenschaften daher dringenden Bedarf an Konvertierungsdiensten, die automatisiert Syntax-Dateien zwischen den verschiedenen Statistikpaketen und Versionen konvertieren können.

5.4.4 Anforderungen aus der Medizin an einen Formatkonvertierungsdienst

In unterschiedlichen Bereichen der Medizin gibt es Anforderungen an einen Formatkonvertierungsdienst in einer LZA-Umgebung im Grid. Im Folgenden werden Anforderungen aus dem Bereich der klinischen Studien und der Biostatistik exemplarisch aufgeführt:

Im Bereich von klinischen Studien (siehe auch Fallstudie Medizin, Deliverable 3.1) werden neben den Studien-Datenbanken auch alle wesentlichen Dokumente der Studie in einem Trial Master File archiviert. Das Trial Master File kann dabei „klassische“ Dateiformate zur Archivierung wie PDF bzw. TIFF, Dateiformate für Rohdaten (ASCII, CSV), Bilddaten wie JPEG, HTML als Dateityp des Internets oder S/MIME als Dateiformat für gesicherte E-Mails umfassen. Bei diesen Dateiformaten ist schon eher davon auszugehen, dass sie auch in Zukunft standardmäßig nutzbar sein werden. Allerdings sind zudem Office-Formate wie z.B. Microsoft Word, OpenDoc oder EPS möglich, bei denen z.T. bereits erhebliche Unterschiede schon alleine bei unterschiedlichen Versionen eines Dateiformats existieren. Weiterhin wird mit DICOM ein spezielles Bildformat in der Medizin eingesetzt, welches einen speziellen Viewer erfordert. Auch bei den Studien-Datenbanken sind ganz unterschiedliche Datenformate möglich. Die Spanne reicht von Bilddaten über PDF und XML bis hin zu

speziellen Datenformaten. Beispielsweise gibt es unterschiedliche Formate zur Beschreibung der Strukturdaten von Proteinen, wie Protein Data Bank (PDB) oder das FASTA-Format. Um sicher stellen zu können, dass der Nutzer ein solches Dokument in seiner Nutzerumgebung öffnen kann, ist eine „Conversion on access“ erforderlich. Die Archivierung in der medizinischen Forschung und der Einsatz von elektronischen Datenarchiven sind noch weitgehend ungelöst. Bisher existiert keine einheitliche LZA-Lösung im medizinischen Umfeld.

In der Biostatistik (siehe auch Fallstudie Biostatistik, Deliverable 3.1) werden insbesondere die Datenformate CSV, TXT, XLS, MySQL-dumps (TXT) und XML verwendet, die als Basisdaten für den Input in die Statistikprogramme R und SAS verwendet werden. Hierzu sind zusätzlich noch die Skripte für die Statistiksoftware relevant. Zur vollständigen und verlässlichen späteren Verifikation ist es notwendig, dass die komplette Statistikumgebung mit archiviert wird, da sich die Rechenweise der Software im Zeitablauf durch Updates und Upgrades ändern kann. Eine Konvertierung der Forschungsdaten ist in diesem Fall nicht notwendig.

6 Provenienzdienst

Provenienz-Informationen sollten in die Metadaten des Forschungsdaten-Repositoriums integriert werden, um Informationen bezüglich der Entstehung und Modifikation der jeweiligen Datenentität für die Nachnutzung der Daten zugänglich zu machen. Der Provenienzdienst wird in WissGrid nur als Konzept entwickelt, da aufgrund seiner Infrastruktur- und Middleware-Nähe eine vollständige Implementierung vermutlich außerhalb der Möglichkeit des Projekts ist. Eine Kooperation mit dem D-Grid-Integrationsprojekt bezüglich dieses Themas wird angestrebt.

6.1 Funktionalität

Für Prozessinformationen im Grid sind zwei Quellen zu unterscheiden: die grid-spezifische Middleware und die ausgeführten Dienste/Jobs.

Für das Grid ist mit dem OGF-Usage-Record ein XML Container zur Beschreibung von Jobs, die auf Grid-Ressourcen ausgeführt werden, definiert worden[11]. Er enthält wesentliche Daten zur Identifikation des Jobs und den Ausführungsbedingungen wie z.B. von wann bis wann er lief und welche Ressourcen benutzt wurden.

Die Prozessinformationen der Dienste selbst sind Teil ihres Outputs. Die Dienste müssen hinreichend aussagekräftige Statusinformationen ausgeben. In dieser Hinsicht kann ein Provenienzdienst nichts leisten, was die Dienste nicht selbst leisten. Als wichtige Anforderung für WissGrid ergibt sich hier allerdings, dass alle Entwicklungen selbst hinreichende Statusinformationen produzieren müssen.

Die Community muss dann die Prozessinformationen, die sie benötigt, auswählen und z.B. in ihr Repository übernehmen. Für eine genauere Bestimmung und theoretische Fundierung, welche Prozessinformationen üblicherweise notwendig sind, wird ein Mapping der Daten des OGF-Usage-Records auf die Event-Metadaten des PREMIS-Standards[12] erstellt. PREMIS definiert für die Langzeitarchivierung notwendige Metadaten und in dem Event-Untertyp die notwendigen Prozessinformationen.

[11] OGF Usage Record – Format Recommendation, http://www.ogf.org/documents/GFD.98.pdf
[12] PREMIS Editorial Committee (Hrsg.): Data Dictionary for Preservation Metadata: PREMIS version 2.0, http://www.loc.gov/standards/premis/v2/premis-2-0.pdf

6.2 Technische Umsetzung

Für die technische Umsetzung eines Provenienzdienstes zur Erzeugung eines OGF Usage Records bei Ausführung von Jobs muss eine Middleware-Ergänzung entwickelt werden, die vor und nach Ausführung die entsprechenden Metadaten erfasst. Weiter zu untersuchen ist, ob der Usage Record als Zusatzdatei zu der Job-Ausgabe oder in ein separates Verzeichnis vorübergehend gespeichert werden sollte.[13] Abbildung 4 gibt dazu eine schematische Darstellung.

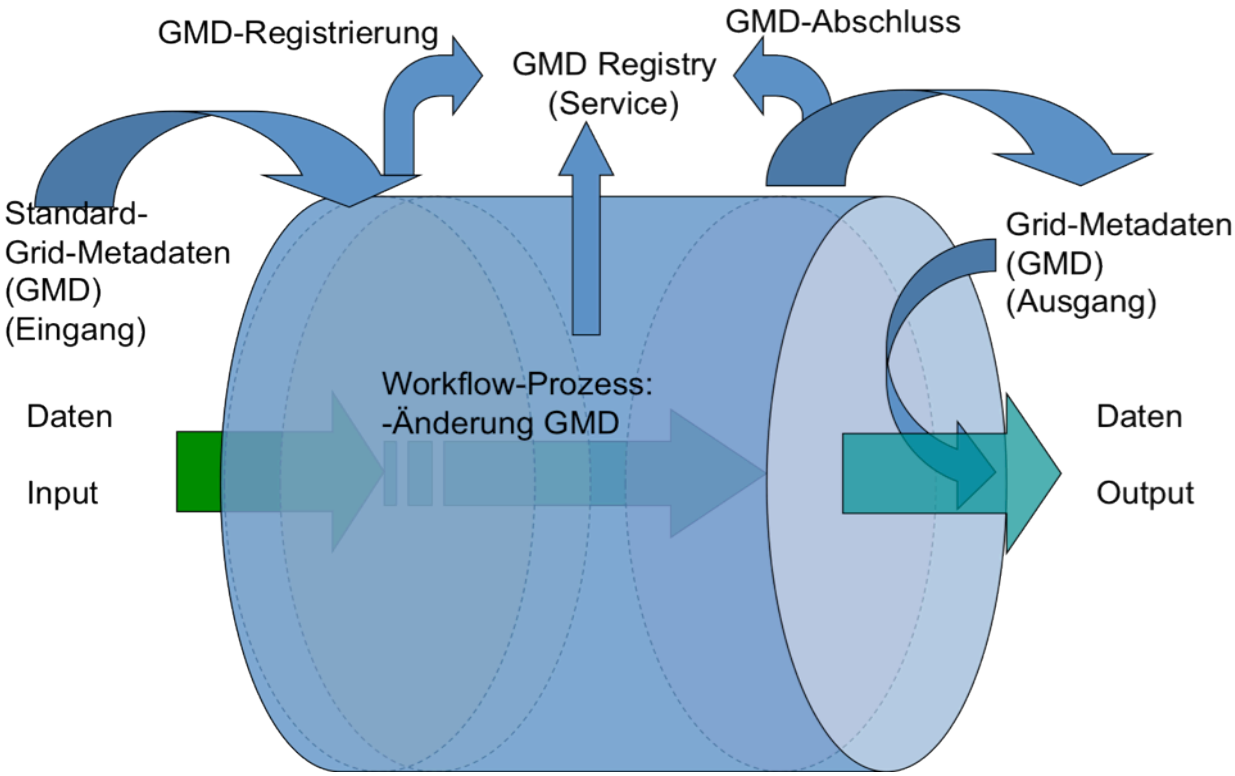


Abbildung 4: Skizze zur Erzeugung von Provenienzdaten

6.3 Entwicklungsaufgaben

- Verhältnis zu SLAs analysieren
- Detailanalyse der Metadatensets: Mapping der Daten des OGF-Usage-Records auf die Event-Metadaten des PREMIS-Standards
- Spezifikation der notwendigen Middleware-Ergänzungen

[13] Für das Konzept eines Provenienzverzeichnisses kann auf Entwicklungen aus den Projekten AeroGrid und "The EU Provenance Project: Enabling and Supporting Provenance in Grids for Complex Problems" http://www.gridprovenance.org/ zurückgegriffen werden.

6.4 Anwendungsfälle

6.4.1 Anforderungen aus der Medizin an einen Provenienzdienst

In unterschiedlichen Bereichen der Medizin gibt es Anforderungen an einen Provenienzdienst für die Anreicherung von Daten mit Informationen zu deren Herkunft in einer LZA-Umgebung im Grid. Im Folgenden werden Anforderungen aus dem Bereich der klinischen Studien und der Biostatistik exemplarisch aufgeführt:

Im Bereich von klinischen Studien (siehe auch Fallstudie Medizin, Deliverable 3.1) soll mithilfe der Langzeitarchivierung die gesamte durchgeführte Studie langfristig gespeichert und nachnutzbar gemacht werden. Die archivierten Dokumente sollen es erlauben, den **Verlauf einer klinischen Studie zu rekonstruieren** und die Qualität der erhobenen Daten und verwendeten Methoden bzw. Verfahren auch nachträglich zu evaluieren. Für diese Rekonstruktion sind die Informationen der Herkunft der Daten (Provenienz) entscheidend. Ein weiterer wichtiger Aspekt ist die **verteilte Verantwortlichkeit** bei der Archivierung. Sowohl der Sponsor als auch der einzelne Prüfarzt haben Archivierungsverpflichtungen. Der Prüfarzt muss persönlich die Aufbewahrung seiner prüfungsbezogenen Unterlagen (Prüfarztordner inkl. der Patientenidentifikationsliste) gewährleisten, der Sponsor die Archivierung seiner gesamten Studienunterlagen (Trial Master File). Durch die unterschiedlichen Verantwortlichkeiten ist eine Unterscheidung der Herkunft der Daten sinnvoll. Zudem besteht in der Nachnutzung die Anforderung der **Revisionsfähigkeit**, welche derzeit mithilfe von digitalen Signaturen gelöst worden ist. Bei elektronischen Dokumenten unterscheidet das Signaturgesetz (SigG[14]) zwischen verschieden Formen von elektronischen Signaturen. Es gibt die einfache elektronische Signatur, die fortgeschrittene elektronische Signatur und die qualifizierte elektronische Signatur.

In der Biostatistik (siehe auch Fallstudie Biostatistik, Deliverable 3.1) kann ein Provenienzdienst die Bereitschaftschaft zur Nutzung des LZA-Dienstes insgesamt erhöhen, indem das **geistige Eigentum** (Intellectual Property Rights, IPR) des einzelnen Forschers geschützt wird. Das Interesse an IPR besteht, da die Forschungsdaten erheblichen Einfluss auf den Erfolg der wissenschaftlichen Arbeit der einzelnen Biostatistiker haben. Der Grund hierfür liegt darin, dass sich aus den Forschungsergebnissen Patente ergeben können und eine Patentierung nur möglich ist, wenn die zugrundeliegende Datenbasis bei Beantragung des Patents nicht öffentlich zur Verfügung steht. Der Provenienzdienst kann somit als Grundlage

[14] Gesetz über Rahmenbedingungen für elektronische Signaturen, http://bundesrecht.juris.de/sigg_2001/index.html

für die Etablierung von Verträgen, welche den Umgang mit den Forschungsdaten regeln. Weiterhin ist eine **Änderungshistorie** zur Nachvollziehbarkeit von Änderungen seitens der Biostatistik notwendig. Die Herkunft der Daten ist dabei wesentlich. Der Einsatz der qualifizierten elektronischen Signatur ist im Bereich der Biostatistik jedoch nicht erforderlich, sofern der Datenbestand der medizinischen Datensätze (MDAT) reinen Forschungscharakter besitzt.

7 WissGrid Dienste Framework

7.1 Funktionalität

Dieser Abschnitt konzentriert sich auf LZA-Dienste, die als Grid-Dienste genutzt werden. Die Nutzung von Web Services für LZA wie dem Planets Interoperability Framework ist auf einer höheren Applikationsschicht anzusiedeln, und deren Integration in das WDF wird derzeit nicht anvisiert.

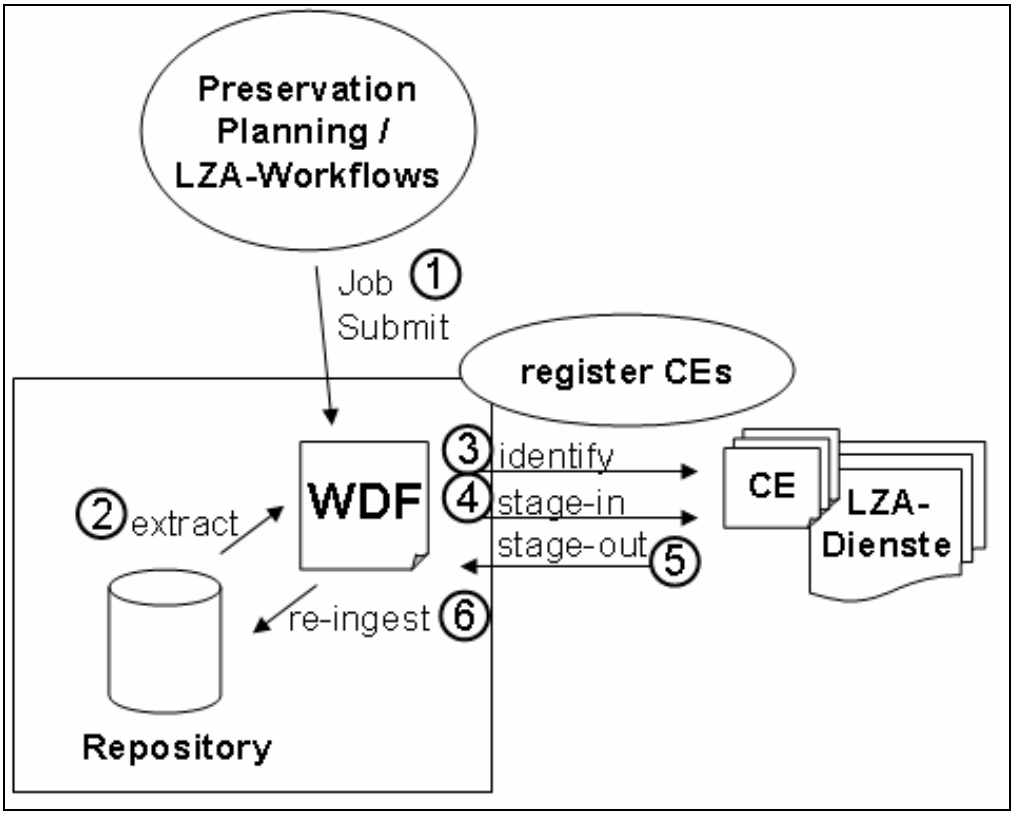


Abbildung 5: Interaktion Dienste-Repository durch WDF

In Abbildung 5 ist die Kommunikation zur Ausführung einer typischen LZA Aufgabe schematisch dargestellt. Die Hauptkomponenten sind dabei

- das Repository, in dem die Daten vorliegen
- der WDF-Dienst, der meist eng mit einem Repository verknüpft ist und auch in einer gemeinsamen Sicherheitsdomäne angesiedelt ist
- der Auslöser einer LZA-Aufgabe, der mehr eine organisatorische als eine technische Einheit ist, und meist durch Menschen gesteuert wird (z.B. die Preservation-Planning-Einheit eines Forschungsarchivs)

- die LZA-Dienste, die auf Computing Elementen (CE) installiert sind und über Grid-Mechanismen angesprochen werden können

[Beschreibungen von LZA-Diensten und zugehörigen CEs werden explizit beim WDF zur Nutzung registriert]

Der Ablauf einer LZA Aufgabe wird über das WDF koordiniert und beinhaltet folgende Schritte:

1. Submit: Annahme der LZA-Aufgabe (z.B. Konvertiere alle TIFF Daten, die älter als 2 Jahre sind, in ein JPEG 2000 Format)
2. Extract: Filterung der gefragten Daten aus dem Repository
3. Identify: Auswahl einer verfügbaren CE, auf der der gefragte Dienst installiert ist
4. Stage-In: Transfer der Daten (im Batch) auf das CE
5. Ausführung des Dienstes
6. Stage-Out: Annahme der konvertierten Daten, nach Ablauf des Jobs; ggf. Zwischenspeicherung der Daten
7. ggf. Re-Ingest: Rückführung in das Repository, gegebenenfalls unter neuerlicher Vergabe von Metadaten und Erstellung einer neuen Objekt-Version

7.2 Architektur

In dieser ersten Version der Spezifikation beläuft sich das folgende Kapitel auf eine vorläufige Skizze, die im weiteren Projektverlauf fortgeschrieben wird.

Der Einsatz des WDF ist an mehreren unterschiedlichen Stellen denkbar:

- aus der Administration des Repositories (Archival Information Update, OAIS) heraus, wenn eine Routine LZA-Aufgabe im laufenden Betrieb erfüllt werden muss, z.B. wenn alle Objekte in einem veralteten Format in ein aktuelles Format migriert werden
- bei der Einfuhr neuer Objekte in das Repository (Ingest, OAIS), z.B. zur Validierung von Dateiformaten
- beim Zugriff auf Repository Inhalte (Access, OAIS), z.B. bei einer Migration-on-Access LZA-Strategie, bei der die Inhalte als Originale oder in einem Standard Format im

Repository vorgehalten werden, und nur beim Zugriff in ein vom Nutzer gewünschtes Format überführt werden

- für Nutzer-gesteuerte Workflows, in denen LZA-Dienste benötigt werden

Um diese generische Einsetzbarkeit zu erreichen, ist das WDF als unabhängige Komponente konzipiert, die aber physisch "nahe" von einem Host-Repository installiert wird: (a) um maximale Datenübertragungsraten für die oft umfangreichen zu transportierenden Datenmengen zu erreichen, und auch (b) um ein Repository und ein zugehöriges WDF in einer gemeinsamen Sicherheitsdomäne zu behalten und damit Rechtefragen zu vereinfachen. Somit wird perspektivisch für jedes Repository ein WDF installiert.

Um dem WDF die Kommunikation mit verschiedenen Repositories zu erlauben, die auf unterschiedlichen Software-Plattformen basieren, ist die Definition der Schnittstelle zwischen dem Repository und dem WDF (siehe Repository Spezifikation, Kapitel 2), wie auch die Schnittstelle zwischen dem WDF und den Diensten als Standard zu definieren.

Die Registrierung der verfügbaren LZA-Dienste auf den CEs wird im Rahmen des WissGrid Projektes voraussichtlich lediglich hart-kodiert. Perspektivisch wird die Architektur so offen ausgelegt, dass die vom WisNetGrid-Projekt geplante Service Registry an das WDF angedockt werden kann. Geprüft wird auch, ob CEs und ihre zugehörigen LZA-Dienste einer expliziten "Trust Zone" (z.B. öffentlicher Grid-Server vs. dezidierter LZA-Server mit starken Sicherheitsrestriktionen) zugewiesen werden müssen, um das Vertrauen der Nutzer zu stärken.

Für die nächsten Schritte werden zunächst folgende einschränkende Annahmen getroffen:

- LZA-Dienste

  Als LZA-Dienst wird zunächst JHOVE (JSTOR-Harvard Object Validation Environment, hul.harvard.edu/jhove) für die Formatcharakterisierung und -validierung exemplarisch hergenommen.
- Repositories

  Als Repository wird zunächst iRODS (Integrated Rule-Oriented Data System, www.irods.org) exemplarisch hergenommen.
- Daten werden zu den Diensten transportiert.

  Im Gegensatz zu dem Ansatz Dienste zu Daten erlaubt dieser Ansatz eine Lastverteilung auf die zur Verfügung stehenden CEs. Zudem verarbeiten die Dienste

die erhaltenen Daten ohne zusätzlich Rechte bezüglich des Datenzugriffs überprüfen zu müssen. Allerdings muss bei der Strategie „Daten zu Diensten“ mit hohen Datentransportlasten gerechnet werden. Daher sollte in einer nächsten Stufe auch die Möglichkeit „Dienste zu den Daten“ untersucht werden.

- Das WDF soll die Zugriffskontrolle sichern.

  Das WDF ist als unabhängige Komponente in der Lage abzusichern, dass auf die Daten, die an Dienste zur Verarbeitung geschickt werden, auch ein berechtigter Zugriff erfolgt. Es stellt sicher, dass automatische Workflows unter Berücksichtigung der Zugriffsrechte auf Daten vollständig abgearbeitet werden können.
- Für jedes Repository wird ein WDF installiert.

  Die einzelnen Repositories haben unterschiedliche Sicherheitsdomänen, so dass ein gemeinsames WDF zu Problemen führen würde. Es ist zu klären, ob es so generisch sein kann, dass es für unterschiedliche Repositories einsetzbar ist.
- Daten werden nur aus einem Repository verarbeitet und gegebenenfalls in dieses wieder zurückgeschrieben.

  Szenarien, in denen Daten aus einem oder mehreren Repositories gelesen bzw. verarbeitet werden und in einem oder mehreren Repositories abgelegt werden, würden in den Bereich föderierte Repositories fallen und sind hier vorerst von der Betrachtung ausgeschlossen.
- Das WDF soll aus unterschiedlichen Anwendungsszenarien heraus agieren können:
  - Beim Einfügen neuer Objekte
  - Aus einem Workflow heraus (bzw. durch den Nutzer)
  - Beim Zugriff auf Daten

  Anders ausgedrückt: Die CRUD-Funktionalitäten sollen wie folgt nutzbar sein:

  - Einzeln
  - Aus einem Workflow heraus

    In der E-Science erfreut sich Taverna, www.taverna.org.uk, für die Spezifikation und Ausführung von Workflows, einer großen Beliebtheit. Es ist sicherzustellen, dass Taverna eingebunden werden kann.
  - Im Rahmen von Datenmanagement-Routinen

Die unteren beiden Schichten von WisNetGrid, Daten und Informationen, sollen insbesondere die

- Bereitstellung von Diensten zur Integration und Vernetzung von Daten und zugehörigen Metadaten-Ressourcen
    - Metadaten- und Ontologiedienste
    - Verzeichnisse und Workflows

sowie die

- Einbindung von Grid-Datendiensten über der technischen Schicht der gemeinsamen Hardware- und Middleware-Ressourcen

gewährleisten.

Im Sinne der D-Grid-Philosophie (keine Eigenentwicklungen, wenn Wiederverwendbarkeit möglich ist) wird sich WissGrid mit WisNetGrid in Verbindung setzen und klären, inwieweit das WDF zunächst mit den obigen Anforderungen unter den angegebenen Einschränkungen mit Methoden von WisNetGrid realisierbar ist.

- Der Einsatz von OSGI (www.osgi.org), einer Softwareplattform zum Bauen und Verwalten von Anwendungen, ist in der Digital Library Community weit verbreitet. Es ist zu untersuchen, in wie weit hierauf zurückgegriffen werden kann.

7.3 Entwicklungsaufgaben

- Kommunikation mit WisNetGrid-Projekt und Evaluierung ihrer Technologien
- Falls WisNetGrid-Technologien sinnvoll sind, aber mit dem WissGrid-Zeitplan nicht zusammenpassen, dann eine Übergangs- oder Alternativlösung festlegen (z.B. fest kodierte Workflows, Skripte, etc)
- Implementation
- Interaktion von Repositorien mit Diensten testen
- Anleitung schreiben
- Installationspaket erstellen

8 Anhänge

8.1 Anhang 1: J-NetCDF Metadata Extractor

JaNEME oder der **J-NetCDF Metadata Extractor** ist ein unter BSD-Lizenz freigegebenes Jhove2-Formatmodul, das am Alfred-Wegener-Institut für Polar- und Meeresforschung entwickelt wurde. Dieses Modul kann Dateien im NetCDF-Format der Versionen 3.0 und 4.0 durch den Einsatz der Unidata java-netcdf Bibliothek 4.0.4 parsen und charakterisieren. Mit den durch diesen Extraktionsprozess gewonnenen Metadaten kann JaNEME die mitgelieferten Templates ausfüllen, die zu dem C3Grid-Profil des ISO 19115/139 und Dublin Core konform sind.

JaNEME basiert so wie die ganze Jhove2-Applikation auf dem Spring Framework 2.5. Dies unterstützt die Verbreitung und spezifische Anpassung des Quellcodes, reduziert die Größe des Codes, steigert die Testbarkeit und Erweiterbarkeit, vereinfacht die Verwaltung der Abhängigkeiten und verbessert deutlich das Applikationsdesign dank der in Spring verfügbaren Designpatterns. Die grundlegende Konfiguration JaNEMEs ist in einer XML-Datei für ihren Einsatz in dem IoC Container erhalten.

In der derzeit verfügbaren Version können nur einzelne Dateien prozessiert werden; Verzeichnisse oder Aggregate werden unterstützt. Komprimierte Dateien (zip, tar.gz?) sind vor der Metadatenextraktion zu entpacken.

Unter http://aforge.awi.de/gf/project/jhove2/frs/ ist das aktuellste Candidate-Release 1.1 JaNEMEs öffentlich herunterladbar.

8.1.1 Projekt-Abhängigkeiten

Neben der Abhängigkeiten Jhove2's werden die auf der untenstehenden Tabelle erläuterten Bibliotheken für seine Kompilierung durch Maven und Exekution benötigt.

Maven ArtifactId	Maven GroupId	Version
netcdf-java	essi-unidata	4.0.41
velocity	org.apache.velocity	1.6.2
velocity-tools	org.apache.velocity	2.0-alpha1
derby	org.apache.derby	10.5.3.0_1
mail	javax.mail	1.4.1
spring-jdbc	org.springframework	2.5.3
spring-aop	org.springframework	2.5.3
cglib	cglib	2.2
asm	asm	3.2
simple-log-slf4j	org.grlea.log.adapters	2.0.1

Tabelle 1: Abhängigkeiten

Damit der automatische Download der netcdf-java Bibliothek in Maven erfolgt, wurde die folgende Remote-Repository-Deklaration in die *pom.xml* Datei des Projektes hinzufügt.

```
<project>
...
    <repositories>
        <repository>
            <id>netcdf-java</id>
            <url>http://maven.generationcp.org/m2po/</url>
        </repository>
```

</repositories>

</project>

8.1.2 Quellcode-Architektur

Der Quellcode und seine JUnit tests liegen in den Packages *src/main/java* bzw. *src/test/java*. Alle dazugehörigen Ressourcen findet man im Verzeichnis *src/main/resources/netcdf.*

Die Registrierung des NetCDF-Moduls im Jhove Namespace erfolgt durch das Einfügen des folgenden Textes in die Property-Datei *src/main/resource/properties/dispatcher.properties*.

info:\:jhove2/format/netcdf NetCDFModule

Die nötigen Beans, ihr Geltungsbereich, die gegenseitigen Abhängigkeiten und ihre zugehörigen Ressourcen werden in *src/main/resources/netcdf/netcdf-config.xml* nach der Spring 2.5 Syntax deklariert.

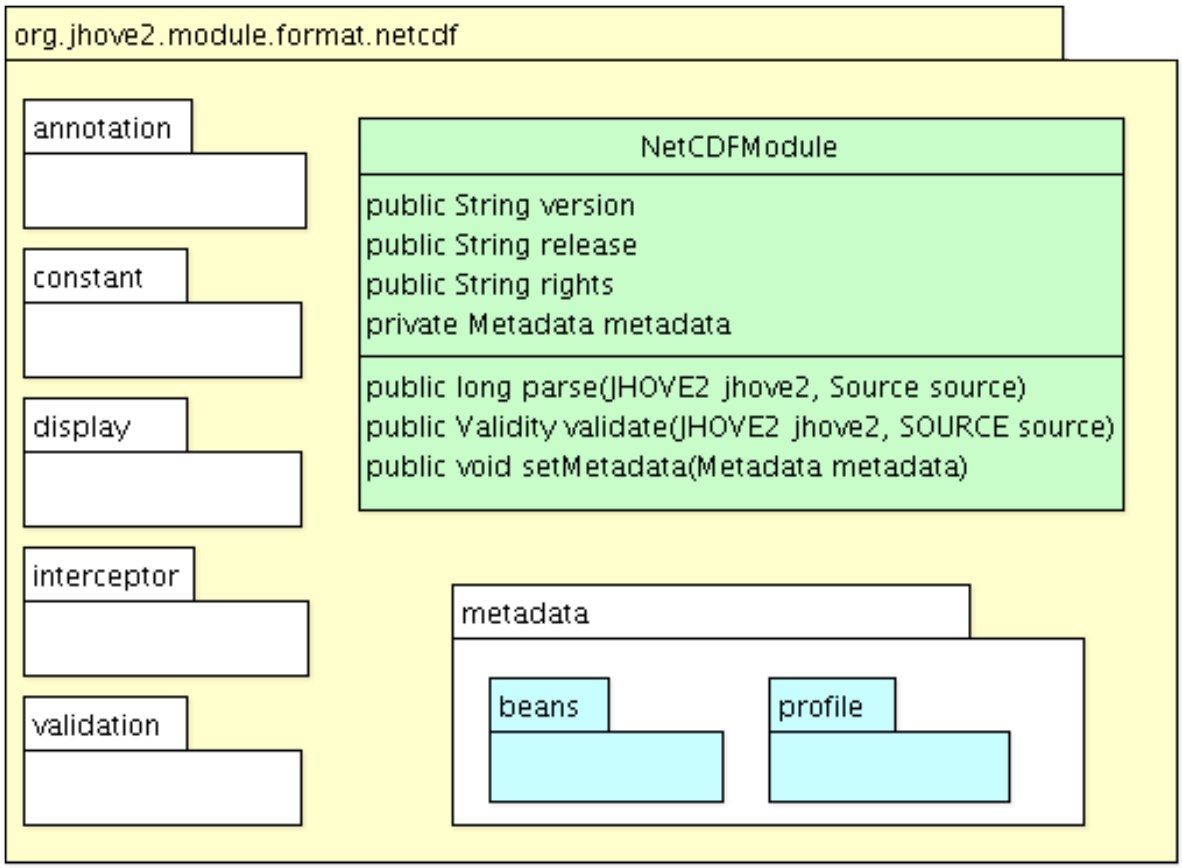


Abbildung 6: Package Diagram

In der Abbildung 6 ist die Organisation des Codes schematisch dargestellt. Im Package *org.jhove2.module.format.netcdf,* findet man die Klasse *NetCDFModule* – das Herz JaNEMEs – und ihre anverwandten Subpackages.

Das Package *metadata* besteht aus den Subpackages *beans* und *profile*. Die von dem Fileheader extrahierten Attributwerte werden in einem *Metadata* Bean (aus dem Subpackage *beans*) gespeichert, dessen Instanz in das *NetCDFModul* durch eine Spring-Bean-Referenz-Deklaration angelegt wird.

Die Implementierung enthält zwei Metadatenprofile, nämlich *C3Profile* und *DCProfile.* Die beiden befinden sich im Subpackage *profile* und sind für die Generierung der Metadatendarstellung durch das Ausfüllen eines *Apache Velocity* Templates mit den in *Metadata*-Beans gespeicherten Attributenwerten verantwortlich.

Die Profile können nur die Standardausgabe durch einen *AbstractDisplayer* erreichen. Für diesen Zweck liefert JaNEME beispielhaft die Displayer *C3ProfileDisplayer* (identifiziert als *c3grid*) und *DCProfileDisplayer* (identifiziert als *dublincore*) im Subpackage *display* mit. Der Endbenutzer kann dennoch die Displayer *JSON*, *Text* oder *XML* weiterverwenden.

Anpassungen an andere Metadatenprofile sind leicht möglich. Die Anwendung von JaNEME ist damit flexibel, der Nutzer entscheidet zur Laufzeit, auf welches Metadatenprofil er die extrahierten Metadaten mappen will.

So kann mit den Aufrufen

./jhove2.sh -d c3grid -o output.xml /home/user/Beispiel.nc bzw.

./jhove2.sh -d dublincore -o output.xml /home/user/Beispiel.nc

die Metadateninformation des gleichen Files Beispiel.nc einmal im Hinblick auf das C3Metadatenprofil und einmal auf Dublin Core extrahiert werden.

Darüber hinaus werden die Templates, die Profile sowie die Displayer in der *netcdf-config.xml* Datei eingetragen.

Allgemeine Konstanten wie *keyword_delimiter*, *default_buffersize* und das enum *variable_recognized_attributes* werden im Interface *Constant* aus dem Package *constant* abgelegt.

Im Package *annotation* befindet sich der Quellcode für die Methoden-Annotationen *@RequiredProperty* und *@Validate*. Die erste wird bei Bean-Getters verwendet und liefert bei der Rückgabe eines Null-Wertes eine Warnung. Wird ein String-Returntyp erwartet, so

wird *null* durch „Please insert a value“ in der Ausgabe ersetzt. Die zweite Annotation ist für Setters und zeigt an, dass der Setter-String-Parameter nach einer Validierungskette verifiziert sein soll.

Die Subpackages *interceptor* und *validation* bieten eine Verifikationsfunktionalität für Beans-Properties annotiert mit *@RequiredProperty* oder *@Validate*. Diese Eigenschaft wird im folgenden Abschnitt ausführlicher diskutiert.

Das Standard JaNEME Mapping in *src/main/resources/netcdf.properties* definiert die Beziehungen zwischen den NetCDF globalen Attributen und den Bean-Properties durch Ausdrücke der Art

netcdf.metadata.{Attribut_name}= {Setter-Ausdruck relativ zu dem Metadata Bean}Als ein Beispiel würde die Definition

netcdf.metadata.acknowledgment=documentation.acknowledgment

bedeuten dass das Attribut “acknowledgment” in Metadata.getDocumention.setAcknowledgment() zu speichern ist.

Dieses Mapping kann beliebig modifiziert werden, wobei dazu neu compiliert werden muss. Als Alternative dazu bietet JaNEME dem Endbenutzer die Möglichkeit, das Mapping durch eine Property-Datei zu steuern, deren Pfad in der System-Property *netcdf.mapping* angegeben werden muss (Beispiel: *JAVA_OPTS=“-Dnetcdf.mappings=/home/user/myMapping“).*

In diesem individuell anpassbaren File kann dann auch das Schlüsselwörtertrennzeichen als regulärer Ausdruck mittels *netcdf.metadata.delimiters=[,\n]+* festgelegt werden. Sollte die Property *netcdf.mappings* nicht definiert sein, wird das standard Mapping verwendet.

Am Ende der c3grid-Profil-Ausgabe werden all die Attribute und ihre Werte als Kommentar aufgelistet, für die kein Mapping definiert wurde. Dies ermöglicht Korrekturen der Ausgabe durch XSLT Transformationen oder weist darauf hin, dass Anpassungen in dem Mapping nötig sein könnten.

8.1.3 Validierungsfunktionalitäten

Mit Hilfe der AOP (Aspect Oriented Programming) Unterstützung ist eine Validierung in JaNEME möglich. Die Klasse *ValidateAroundAdvice* aus dem Package *validation* verhält

sich wie ein Interceptor für Getters und Setters in den Metadatenbeans, deren Instanzen durch Proxies gehandelt werden. Durch die Deklaration eines *BeannameAutoProxyCreator* 's wird dieses Verhältnis dem IoC Container mitgeteilt.

Die Methode *invoke* der Java-Klasse *ValidateAroundAdvice* kann vor und nach dem Aufruf der Bean-Target-Methode ablaufen. So kann überprüft werden, ob die begleitenden Annotationen *@RequiredProperty* und *@Validate* existieren, und ein entsprechender Validierungsprozess kann durchlaufen werden.

Die Annotation *@Validate* präzisiert den Namen einer Validierungskette, der mit dem *id* einer ValidationChain-Bean in der Spring-Konfiguration *netcdf-config.xml* übereinstimmt. Eine Validierungskette besteht aus einer ArrayList von Validatoren und einer *String*-Nachricht für den Fall, dass die Validation scheitert – wenn die Methode *isValid* der ValidationChain *false* zurückgibt. Dies ermöglicht das Testen der Konformität von Variablennamen mit einem bestimmten Vokabular, z.B. der CF Convention, in einer spezifischen Reihenfolge. Immer wenn die Methode *isValid* eines Validators *true* zurückgibt, sind keine weiteren Verifikationen mehr nötig. Bei einem Fehlschlag eines Validators wird eine weitere Verifikation durch den nachfolgenden Validator in der Kette erzwungen. So kann eine Kette *false* dann und nur dann zurückgeben, wenn alle ihre Validatoren *false* zurückgeben.

Ein Vorteil der Validierungskette besteht in der Wiederverwendung der Validatoren in verschiedenen Ketten.

Die Klassen für *ValidationChain*, Validatoren und ihre unterstützenden Klassen befinden sich im Subpackage *validation*.

Für die Verbreitung des Moduls ist wesentlich, dass die Erweiterung einer Validation in JaNEME konzeptionell sehr trivial ist. Die weitere Entwicklung eines vorhandenen Vokabulars sowie die Integration eines neuen in das Modul sind möglich. Dieses Vokabular kann in einem enum, array oder sogar in einer Datenbank gespeichert werden. Exemplarisch ist das für den CFValidator, der eine portable Apache Derby Datenbank bestehend aus 1907 CF-Standardnamen und 145 Aliases anfragt.

8.1.4 Einschränkungen

Das erste Release JaNEMEs wurde an der Alpha jhove2-0.5.2.zip angebunden. Um mögliche Kompilationsprobleme zu umgehen, ist in dem oben aufgeführten Verzeichnis am AWI das Gesamtpaket Jhove und JaNEME enthalten. Mit den Entwicklern von JHove2 ist aber bereits Kontaktaufgenommen worden. Sie begrüßen sehr die hier vorstellten Entwicklungen und

erachten sie als so wichtig, dass über eine Integration von JaNEME in die offizielle Jhove Distribution nachgedacht wird.

Derzeit gibt es drei wesentliche Limitierungen beim Einsatz von JaNEME. So ist bei Modifikationen im Quellcode unbedingt Eclipse als Entwicklungs-IDE erforderlich.

Die beiden anderen Limitierung sind inhaltlicher Art: Jhove2 enthält im Vergleich zu Jhove1 bedeutend weniger Formatmodule. Daher werden derzeit alle NetCDF-Dateien (außer zip) als UTF-8 identifiziert. Dafür wurde ein Workaround eingesetzt, indem in der Klasse *org.jhove2.module.identify.IdentifierModule* folgende Ersetzung vorgenommen wird:

id= new Formatidentification(Configure.getReportable(Format.class, „UTF8Format“), Confidence.PositiveGeneric, this.wrappedProduct);

durch

id= new FormatIdentification(Configure.getReportable(Format.class, „NetCDFFormat“), Configende.PositiveGeneric, this.wrappedProduct):

Da es keine Standards für das Mapping von NetCDF auf ISO 19115 oder Dublin Core gibt, muss der Endbenutzer eventuell die entsprechenden Templates oder sogar den Quellcode anpassen.

8.2 Anhang 2: Glossar

Bitstream Preservation: Bitstream Preservation ist eine Form der Langzeitarchivierung, die darauf zielt, dass jedes Bit eines Datenobjekts ohne unbeabsichtigte Veränderungen verfügbar ist. Sie begegnet so zum Beispiel dem Verfall der Speichermedien und Speichertechnologien und beinhaltet Aktivitäten wie regelmäßige Integritätstests und das Anlegen von verteilten und unabhängigen Kopien. In einem Grid-Umfeld ist die Schaffung spezifischer "Trust Zones" denkbar, in denen ausgewählte Storage-Ressourcen besondere Qualitäts- und Sicherheitskriterien erfüllen. (Siehe auch D3.1 "Generische Langzeitarchivierungsarchitektur für D-Grid", Anhang 3.)

Charakterisierung: siehe Formatcharakterisierung

Content Preservation: Content Preservation ist eine Form der Langzeitarchivierung, die darauf zielt, die technische Nutzbarkeit von Daten zu erhalten. Sie umfasst Aktivitäten wie eine kontinuierliche Beobachtung der Technologieentwicklung, technische Qualitätskontrollen und Erhaltungsmaßnahmen wie Formatkonvertierungen/Migrationen oder die Bereitstellung von Emulatoren. (Siehe auch D3.1 "Generische Langzeitarchivierungsarchitektur für D-Grid", Anhang 3.)

CRUD: CRUD steht für Create/Read/Update/Delete und stellt die Grundbausteine von Datenverwaltungssystemen (z.B. Datenbanken und Repositorien) dar.

CSV: Comma Separated Value steht für durch Komma und Zeilenumbrüche definierte zweidimensionale Datenfelder (z.B. Zahlenkolonnen) in ASCII-Dateien. Dieses einfach strukturierte Format ist in RFC 4180 beschrieben. Moderne Anwendungen wie Excel sind nicht auf das Komma als Trennzeichen angewiesen. Der Nutzer kann das Trennzeichen hier vorgeben.

Data Curation: Eine Form der Langzeitarchivierung, die darauf zielt, die intellektuelle Nutzbarkeit von Daten zu erhalten. Sie umfasst Aktivitäten wie die Konzeption von Daten und Metadaten, Versionierung von Objekten, Bereitstellung von notwendigen Hintergrund- und Kontextinformationen, etc. (Siehe auch D3.1 "Generische Langzeitarchivierungsarchitektur für D-Grid", Anhang 3.)

Digitale Objekte: Digitale Objekte sind digitale Daten, die als intellektuelle Einheiten aus (einer oder mehreren) Dateien, zugehörigen Metadaten, sowie einem Netzwerk aus

anderen Objekten bzw. referenzierbaren Informationen bestehen können. Objekte können alle Arten von Daten umfassen - strukturiert, semi-strukturiert (z.B. XML-basiert), oder unstrukturierte Daten wie z.B. Bilder oder Videos. Um sie explizit zu beschrieben, können so genannte Paketformate benutzt werden, die zugehörige Metadaten (z.B. deskriptiv, administrativ, Audit Trails) wie auch Relationen zu anderen Objekten und externen Erschließungsmaterialien enthalten.

Formatcharakterisierung: Ein Dienst zur Formatcharakterisierung extrahiert aus digitalen Objekten Metadaten wie z.B. die eindeutige Bezeichnung des Formats und der Formatversion. Die Metadaten können technischer Natur sein, wie z.B. Auflösung und Farbrauminformationen bei Bildformaten oder Erstellungssoftware und -hardware, aber auch deskriptive Metadaten, die das Objekt intellektuell beschreiben, wenn sie eingebettet sind. Metadaten sind notwendig, um Daten effektiv verwalten und nutzen zu können.

Formatkonvertierung: Dienste zur Formatkonvertierung überführen digitale Objekte von einem Format möglichst verlustfrei in ein anderes. Damit ist es nicht nur möglich wichtige Daten in veralteten Formaten durch die Umwandlung in aktuelle Formate nutzbar zu halten, sondern auch der Datenaustauch kann durch die Anpassung von Daten an fremde Schnittstellen und Software erleichtert werden.

Formatvalidierung: Formatvalidierung ist die Prüfung eines digitalen Objekts auf seine technische Korrektheit, ob die syntaktischen und ggf. semantischen Vorschriften des Formats eingehalten werden, und stellt einen Teil einer Qualitätssicherung dar.

Forschungsdatenarchiv (bzw. synonym "Forschungsarchiv"): Ein Forschungsarchiv umfasst Technik und Organisation (z.B. Betrieb, Finanzierung, Verantwortlichkeiten). Dazu passt es die generischen Funktionalitäten von Repositorien an den spezifischen Kontext einer Community an (z.B. Anwendungsszenarien, organisatorischer Rahmen). Vor allem Vertrauenswürdigkeit und Langzeitarchivierung von Objekten bauen zwar auf die technische Basis von Repositorien und LZA-Diensten, können letztlich aber nur durch darüber liegende organisatorische Maßnahmen gewährleistet werden (z.B. finanzielle Stabilität, Rollen für Preservation Planning und Audit[15]).

Langzeitarchivierung (LZA): Alle Aktivitäten, die darauf abzielen die Nutzbarkeit digitaler Daten angesichts eines sich verändernden Kontextes (zeitlich, technisch, intellektuell, etc.)

[15] Research Libraries Group. (2002). Trusted digital repositories: Attributes and responsibilities. An RLG-OCLC Report. http://www.oclc.org/programs/ourwork/past/trustedrep/repositories.pdf

zu erhalten. Formen der Langzeitarchivierung sind Bitstream Preservation, Content Preservation und Data Curation. (Siehe auch D3.1 "Generische Langzeitarchivierungsarchitektur für D-Grid", Anhang 3.)

Metadatenextraktion: siehe Formatcharakterisierung

NetCDF: Das **Network Common Data Format** ist ein binäres Dateiformat für den Austausch wissenschaftlicher Daten.[16] Das Dateiformat ist selbstbeschreibend – es gibt einen Header, in dem neben Metadaten (in Form von geordneten Paaren aus Schlüsseln und Attributen) auch die Struktur des Datenbereichs beschrieben ist. Die Daten selbst sind als (ein- oder mehrdimensionale) Arrays abgelegt. Dies ermöglicht einen schnellen Zugriff.

Konvertierung: siehe Formatkonvertierung

Provenienzdienst: Ein Provenienzdienst erfasst und speichert Informationen über Prozesse, die Daten verarbeiten und verändern. Er ermöglicht es dadurch die Verarbeitungsschritte langfristig nachvollziehbar zu dokumentieren, um die Authentizität von Daten später bewerten zu können. Es handelt sich dabei um eine Querschnittsfunktionalität, die z.B. in der Middleware implementiert sein muss.

Repository: Softwaresystem zur Verwaltung von digitalen Objekten. Neben der Verwaltung von digitalen Objekten (speichern, abrufen, verändern bzw. neue Versionen anlegen) bieten Repositorien auch zumeist generische Mechanismen zur Einbettung der Objekte in wissenschaftliche und interaktive Workflows, z.B. für die kollaborative Bearbeitung von Objekten in interaktiven Editoren, oder für automatisierte wissenschaftliche Berechnungen.

Trust Zone: Gerade in der LZA ist die Integrität und Sicherheit von Daten oft besonders relevant (siehe Glossar-Eintrag zu "Vertrauenswürdigkeit"). Dies verträgt sich mitunter nicht mit der Offenheit und Verfügbarkeit von Ressourcen in einer Grid-Infrastruktur. Zur Abhilfe können "Vertrauenszonen" (Trust Zone) auf speziellen Grid-Ressourcen geschaffen werden, die durch besondere technische und organisatorische Maßnahmen einen hohen Grad an Vertrauenswürdigkeit sicherstellen können (z.B. Datensicherheit, Datenintegrität, Bit Preservation). Trust Zones können Teil des Konzeptes einer Grid-

[16] http://de.wikipedia.org/wiki/NetCDF

Infrastruktur wie D-Grid sein, können aber technisch auch explizit getrennt von der offenen Compute-Grid-Infrastruktur sein, wo das notwendig ist.

Validierung: siehe Formatvalidierung

Vertrauenswürdigkeit: "Eigenschaft eines Systems, gemäß seinen Zielen und Spezifikationen zu operieren (d.h. es tut genau das, was es zu tun vorgibt bzw. was seine Betreiber versprechen, dass es tut). Aus der Sicht eines Benutzers ist ein System vertrauenswürdig, wenn seine Erwartungen erfüllt werden." (Entnommen aus "Kriterienkatalog zur Prüfung der Vertrauenswürdigkeit von PI-Systemen", nestor (Hrsg.), 2009, urn:nbn:de:0008-20080710140). In der Langzeitarchivierung wird die Vertrauenswürdigkeit üblicherweise durch Kriterien geprüft, die die Wahrscheinlichkeit der Nachnutzbarkeit der Daten erhöhen.

WSDL: Die **Web Services Description Language** ist eine Beschreibungssprache für Netzwerkdienste (Web-Services) zum Austausch von Nachrichten auf Basis von XML. Mit Hilfe von WSDL können die angebotenen Funktionen, Daten, Datentypen und Austauschprotokolle eines Web-Service beschrieben werden. WSDL ist unabhängig von Plattform, Programmiersprache und Protokoll und wurde vom World Wide Web Consortium entwickelt.