WissGrid

Deliverable 2.1.2

Arbeitspaket 2: Blaupausen und Beratung

Evaluation, Dokumentation und Registrierung von höheren Diensten[1]

Deliverable	2.1.2 Registrierung von höheren Diensten
Autoren	Arbeitspaket 2: Blaupausen und Beratung
Editoren	G. Stöckle, H. Enke, Christian Grimme, Florian Schintke, Frank Schluenzen, Felix Lohmeier, Wolfgang Pempe
Datum	04-03-2011
Dokument Version	1.0.1

A: Status des Dokuments

Deliverable 2.1.2, Version 1.0.1, Das Dokument ist durch das Projekt akzeptiert.

B: Bezug zum Projektplan

Im Projektantrag Wissgrid wird der Inhalt des Deliverables 2.1.2 folgendermassen vorgegeben: "Evaluation, Dokumentation und Registrierung von in den Community Grids vorhandenen höheren Diensten zur Schaffung einer Beratungs- und Entscheidungsgrundlage für den Aufbau eines Community Grids".

Mit dem Begriff "höhere Dienste" werden Dienste bezeichnet, die nahe an der Anwenderschicht liegen und sich auf den Nutzer konzentrieren. Diese sind nahezu immer von einer Community implementiert, und sind hierdurch auch abgegrenzt zu den im DGI-Projekt entwickelten und gepflegten

[1]This work is created by the WissGrid project. The project is funded by the German Federal Ministry of Education and Research (BMBF).

grundlegenden Diensten und der Middleware, Es sollen hier Dienste vorgestellt werden, die einen Anwendungs-nahen Einstieg in Grid-Technologie ermöglichen und bestimmte Community-nahe Probleme adressieren..

C: Abstract

This document tries to give an overview over higher services in WissGrid. It is meant as basis for decision-making and consulting activities. Higher services are defined as services close to the application layer, with defined interfaces to applications or GUIs. As a further restriction, services described in this document are developments of a specific scientific community. We try to show similar approaches between different scientific communities.

Each section contains an overview over the Community Grid, followed by a detailed description of the identified services which are adressing problems that other Community grids may face also.

D: Änderungen

Version	Date	Name	Brief summary
0.1.0	10.11.2009	Harry Enke	Erstellung des Ausgangsdokuments
0.2.0	30.04.2010	Gabriel Stöckle & Harry Enke	Dokumentstruktur und exemplarischer Inhalt
0.2.1	18.05.2010	Christian Grimme	Dokumentation C3Grid WSS
0.2.2	27.05.2010	Gabriel Stöckle	Anpassungen, Bezug zum Projektplan, etc.
0.3.1	22.06.2010	Gabriel Stöckle	Dienste in AstroGrid-D
0.3.2	06.07.2010	Florian Schintke	C3Grid DMS, sonstige Korrekturen
0.4.0	06.07.2010	Gabriel Stöckle	Dienste in MediGrid
0.4.1	13.07.2010	Frank Schluenzen	HEP Ergänzungen, sonstige Korrekturen
0.5.0	27.07.2010	Gabriel Stöckle	Dienste in TextGrid, Umstrukturierungen
0.5.1	28.07.2010	Gabriel Stöckle	Korrekturen, Summary
0.5.2	28.07.2010	Harry Enke	Korrekturen, Ergänzungen AstroGrid-D
0.6.0	30.07.2010	Gabriel Stöckle, Harry Enke	Korrekturen, Edits, Einbeziehen von Kommentaren
0.6.1	09.08.2010	Felix Lohmeier, Wolfgang Pempe	TextGrid Formulierungen und Inhalte
1.0.0	20.08.2010	Harry Enke	Korrekturen, Typos
1.0.1	04.03.2011	Gabriel Stöckle	Beschreibung und Referenz zu AstroGridTest verbessert

E:

Inhaltsverzeichnis

1 Einleitung

Projektantrag: Unser Ansatz bestand darin Webseiten der Projekte AstroGrid-D [3], C3Grid [4], MediGrid [5], HEPGrid [6],[7] und Textgrid [8] aus dem D-Grid Call 2 zu sichten und dort identifizierte *höhere Dienste* nach folgenden Gesichtspunkten zu ordnen:

Welche Dienste wurden in den Communities entwickelt? Siehe Tabelle zu Beginn jeden Absatzes.

Ziel des Dokuments ist mit Hinblick auf weitere Verwendung der höheren Dienste in neuen Communities eine Abschätzung zu ermöglichen: *Was lohnt sich zu integrieren, welcher Aufwand müsste getrieben werden, um einen Dienst an eine neue Community anzupassen?*

Dienste, oder Entwicklungen die keine "höheren Dienste" im Sinne von Deliv. 2.1.2 darstellen wurden nicht weiter beschrieben.

1.1 Höhere Dienste im Sinne von Deliverable 2.1.2

Als höhere Diensten werden solche Dienste angesehen, die nahe an der Anwendungsebene liegen. Die dargestellten Dienste stellen eine "höhere" Art der Datenverarbeitung dar, da sie beispielsweise die Daten mehrerer Ressourcen kombinieren, Metadaten neu kombinieren, etc.

Weiter soll ein höherer Dienst neben reinen Nutzerinterfaces auch über Schnittstellen zu anderen Anwendungen verfügen. Ein höherer Dienst im Sinne von Deliverable 2.1.2 soll ein spezifisch in einer Community entwickelter Dienst sein. Dienste die allgemein für eine Middleware oder von und für die gesamte Grid-Community entworfen wurden sollen hier ebenfalls nicht beschrieben werden.

Es sollen die Dienste beschrieben werden, die speziell von einer Community für ihre Anwendung entwickelt wurden und die erweiterbare Schnittstellen und Erweiterungsmöglichkeiten bieten, um die Software als Standarddienst in mehreren Community-Grids zu etablieren. Im Gegensatz zu den im DGI-Projekt entwickelten grundlegenden Rahmenbedingungen und Middleware-Konzepten sollen hier Dienste vorgestellt werden, die für einen schnellen und anwendungsnahen Einstieg in Grid-Technologien in Frage kommen.

Spezieller Augenmerk soll bei der Beschreibung gelegt werden auf:

- *Welche Middleware wird unterstützt/verwendet?*
- *Handelt es sich beim beschriebenen Dienst um eine Community-spezifische Entwicklung?*
- *Welche Technologie wird verwendet, welche Schnittstellen sind vorhanden?*

2 Höhere Dienste in AstroGrid-D

2.1 Überblick

Service	Technologie	Beschreibung
Stellaris	RDF und SPARQL	Metadaten Management Dienst
AstroGrid-D Data Management (ADM)		virtuelle Datei-Verwaltung im Grid
Grid timeline		Web-basiertes Echtzeit-Monitoring
AstroGridTest		Testen von AstroGrid-D Ressourcen
OGSA-DAI Dienste		Datenbank Management

Tabelle 1: Höhere Dienste in AstroGrid-D

Middleware-Extensions aus Deliv. 2.1.1 Als Ausgangspunkt für dieses Dokument wurden die in WissGrid Deliverable 2.1.1 [1] identifizierten Middleware-Extensions

- Stellaris
- Robotic Telescopes Package
- TimeLine
- Grid-RessourceMap
- Stream-Based Data Management
- Grid Application Toolkit (GAT)
- filebasierte Daten/Metadaten-Verwaltung in Globus
- Job-Monitoring mit Globus

unter dem Gesichtspunkt "handelt es sich hierbei um Dienste im Sinne von Deliverable 2.1.2?" betrachtet.

Middleware Als Middleware wird in AstroGrid-D das **Globus Toolkit 4** verwendet, hauptsächlich weil es sich dabei um eines der meistverbreiteten Middleware-Systeme handelt. Weiter wurde die Middleware um Metadaten-Komponenten des Virtual Observatorys und um Eigenentwicklungen erweitert.

Kommentar Die Dienste Gridway, Grid Application Toolkit (GAT) sowie das AstroGrid-D Portal werden in diesem Dokument nicht weiter beschrieben, da es sich bei diesen Diensten nicht um Dienste im Sinne des Deliverable 2.1.2 handelt. So hat beispielsweise das AstroGrid-D Portal bisher keine definierten Schnittstellen. Bei der aufgeführten "filebasierte Daten/Metadaten-Verwaltung in Globus" und dem "Job-Monitoring mit Globus" handelt es sich um zwei nicht Community-spezifische Entwicklungen und werden aus diesem Grund ebenfalls nicht weiter beschrieben.

Neben den klassischen Ressourcen wie Speicher- und Computer-Ressourcen wurden in AstroGrid-D auch Routinen zur Anbindung von Instrumenten als Middleware-Erweiterung entwickelt. Bild 1 zeigt wie Applikationen über verschiedene Dienste in verschiedenen Schichten aufgebaut sind.

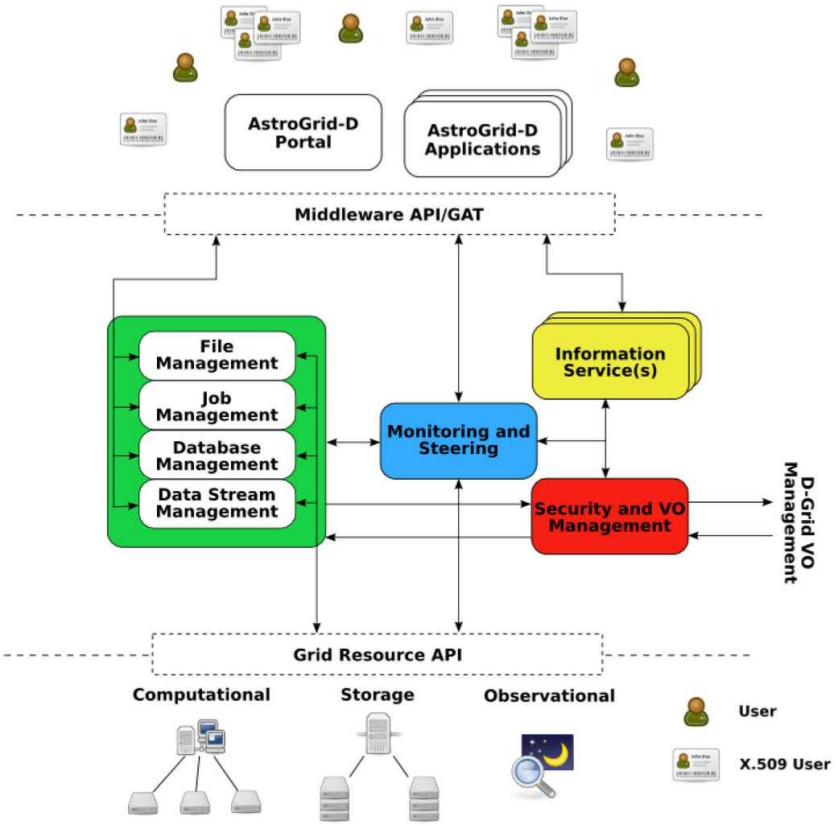

Abbildung 1: AstroGrid-D Architektur, Schichten und Dienste, die an einer Grid-Applikation beteiligt sind. Quelle:[13]

2.2 Stellaris

kurze Beschreibung bei Stellaris handelt es sich um einen Informations-Dienst, der den Metadatenaustausch zwischen einzelnen Ressourcen ermöglicht, im AstroGrid-D zwischen robotischen Teleskopen, für das Speichern von Job-Daten für den Nutzer sowie für Nutzungsdaten der Ressourcen, einen sogenannten *Metadaten Management Service*.

Hierbei ist nicht nur die Möglichkeit Daten und Softwaretools aus dem Grid direkt zu nutzen von Vorteil, sondern vor allem auch die zentrale Grid Nutzer- und VO-Verwaltung des Grid erweist sich als wertvolle Lösung für das zentrale Management von Zugangsrechten und Beobachtungszeiten. Man kann sagen Stellaris verwendet verschiedene Komponenten des Grid als *Information Provider*.

Quellen: [10], [11], [12].

beispielhafte Anwendung/Usecases Zustandserfassung robotischer Teleskope, Job bezogene Grid-Nutzungsdaten (Darstellung durch Timeline),

Technologie RDF und SPARQL, XSLT-Transformation von Globus MDS4 und Usage Record (XML $\leftarrow$ XSLT)

Schnittstellen Messinstrument (robotische Teleskope) - GUI, VO, Stellaris

2.3 Datenbank Management

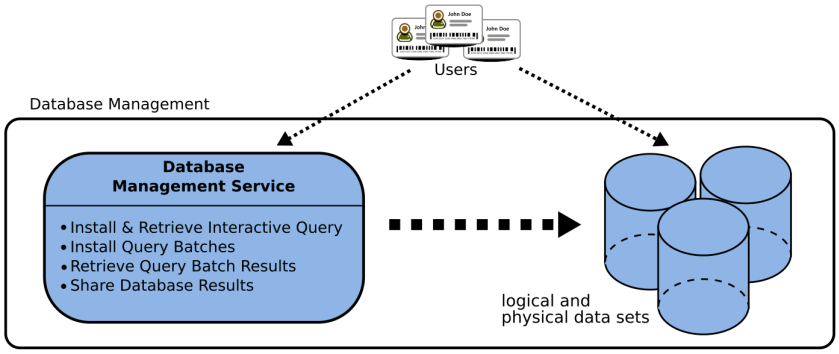

Abbildung 2: AstroGrid-D Database Management. Benutzer können auf Daten direkt oder per batch job zugreifen. Der Ort der Daten im System bleibt transparent. Quelle:[13]

kurze Beschreibung Für den Datenbankzugang (siehe Bild 2) innerhalb von AstroGrid-D wurden für einige wissenschaftliche Projekte Grid-Lösungen aufgrund der räumlichen Verteiltung der Daten sowie der ebenfalls räumlich verteilten Verarbeitung der Daten als sinnvolle Ansätze erkannt. Es wurden hierzu die von OGSA-DAI entwickelten Dienste als Teil der Globus Middleware verwendet. Verschiedene Loadbalancing-Techniken wurden getestet und entwickelt.

beispielhafte Anwendung/Usecase Clusterfinder: bei der Suche innerhalb der SDSS und ROSAT Datenbanken werden durch die Identifikation von Querverbindungen Galaxienhaufen identifiziert. Diese Suche konnte von den entwickelten Loadbalancing-Techniken stark profitieren. Die Datenbank-Verknüpfungen folgen einem fixen Schema, welches auch über die Metadatenbeschreibung des Datenbanksystems zur Verfügung steht.

Technologie Datenbankabfragen werden mit der standardisierten Abfragesprache SQL durchgeführt. Der Einsatz der um fachspezifische Objekte erweiterten und auf SQL basierenden Daten-Abfrage durch die Virtual Observatory Query Language (VOQL, vormals ADQL=Astronomical Data Query Language) zur Datenextraktion im Grid wird vorbereitet.

Schnittstellen VO - SQL-Datenbank

2.4 Datenstrom-Management

kurze Beschreibung Desweiteren wurde ein Daten Stream Management-Modell (siehe Bild 3) erarbeitet. Datenströme entstehen vor allem bei kontinuierlich messenden Ressourcen wie beispielsweise Teleskopen und generieren ständig neue Daten. Wichtigste Vorraussetzung zur Verarbeitung solcher Datenströme ist Datenfilterung. Dieser Bedarf wurde erkannt, ebenso wurde festgestellt, dass momentane Middleware-Strukturen keine Möglichkeit einer derartigen Daten-verarbeitung bieten. Hier wurde das Konzept der Operatoren entwickelt, diese filtern Datenströme innerhalb des DatenGrids an mehreren Stellen. Operatoren können (z.B. über persönliche Webseiten) veröffentlicht werden, in Online-Libraries abgelegt und wiederverwendet werden.

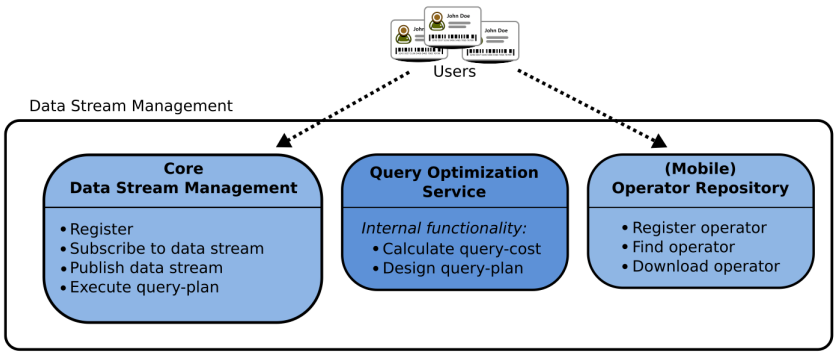

Abbildung 3: AstroGrid-D Data Stream Management. Benutzer können im Daten Strom veröffentlichen und sich für Daten Ströme registrieren, sowie Datenstrom-bezogene Operatoren durch Operator-Repositories teilen. Intern bietet der Data Stream Service Opimierungspotential. Quelle:[13]

Schnittstellen Ressourcen - Operatoren (Webseite)

2.5 AstroGrid-D Data Management (ADM)

kurze Beschreibung Virtuelles Filesystem im Grid. ADM wurde entwickelt als Werkzeug zum verteilten Daten-Management. Per Kommandozeile, per Webinterface oder per Application Interface kann ein Daten-Zugriff erfolgen. Es löst das Problem der Datenzuordnung im Zusammenhang mit der nicht durch den Benutzer kontrollierbaren Job- submission durch den GridWay-Service, speziell die Zusammenarbeit mit dem Globus Replica Location Service (RLS).

beispielhafte Anwendung/Usecase ADM stellt ein Tool zur eindeutigen Identifikation von Dateien und Ordnern im Grid dar, es erstellt automatisch lokale Dateiverweise und markiert diese mit Meta-Markierungen die unabhängig vom jeweiligen den Job ausführenden System sind. Dies ist vor allem nützlich bei der Zusammenstellung von Daten für eine Verarbeitung im Grid und um diese im Anschluss weiter zu verarbeiten.

Technologie ADM basiert auf einer relationalen Datenbank, um eindeutige Datenzuordnung über einen Dateiendescriptor zu ermöglichen. ADM bietet eine client-Software, eine C-library, die den API-Datenzugriff ermöglicht, sowie einen Web-client. Quelle: [13], [3]

Schnittstellen Filesystem-Client, API

2.6 Grid timeline

kurze Beschreibung Echtzeit-Darstellung von mehreren Grid-basierten Jobs. Wird beispielsweise von der Extension *Telescope Timeline* als User Interface bei der Steuerung und Überwachung von robotischen Teleskopen, die in diesem Fall eine Grid-Ressource darstellen, verwendet. Quelle: [3], [13]

beispielhafte Anwendung/Usecases Usage Records der AstroGrid-Nutzer (auf AstroGrid-Ressourcen) Zustandserfassung robotischer Teleskope

Technologie Die Ressourcen-Informationen werden über den Informationsdienst Stellaris per SPARQL Queries abgefragt.

Schnittstellen Stellaris - GUI (Telescope Timeline)

2.7 AstroGridTest

kurze Beschreibung Eine Zusammenstellung von mehreren Skripten und zweierlei grafischen Benutzeroberflächen. Die Programme wurden in den Jahren 2009 bis 2011 von Klaus Rieger in der Skriptsprache *Bash* und in der Programmiersprache *Java* geschrieben. Ihre Aufgabe ist die verschiedenen Hosts zu überprüfen und weitere Informationen über das Grid zu erhalten – insbesondere innerhalb der AstroGrid-D Community.

Quelle: [9]

Technologie Bash-Skripte, Java

Schnittstellen Ressourcen des Grid, GUI (Java, Web)

3 Höhere Dienste in C3Grid

3.1 Überblick

Service	Technologie	Beschreibung
DIS	MDS-Erweiterung	Verwaltung der Metadaten- und Datenressourcen
WSS		Workflow Scheduler
GNDMS		Grid Datenmanagement
Metadatensuchdienst		Metadatensuchdienst

Tabelle 2: höhere Dienste in C3Grid

Middleware-Extensions aus Deliv. 2.1.1 Als Ausgangspunkt für dieses Dokument wurden zusätzlich zu den C3Grid-Web-Seiten die in WissGrid Deliverable 2.1.1 [1] identifizierten Middleware-Erweiterungen

- Portal
- MDS-Erweiterung zur Verwaltung der Metadaten- und Datenressourcen, Eingenentwicklung Dateninformationsdienst (DIS)
- Workflow Scheduling Service (WSS)
- Datenmanagement Service (GNDMS)

unter dem Gesichtspunkt "handelt es sich hierbei um Dienste im Sinne von Deliverable 2.1.2?" betrachtet.

Middleware Als Grid-Middleware wurde das **Globus Toolkit 4** verwendet, da es das meistverbreitete Grid-Middleware-System ist.

Kommentar Das C3Grid-Portal wird in diesem Dokument nicht weiter beschrieben, da es sich dabei nicht um einen Dienst im Sinne des Deliverable 2.1.2 handelt.

3.2 Dateninformationsdienst (DIS)

Kurze Beschreibung: aufbauend auf einer gemeinsamen Definition eines einheitlichen Metadatenprofils mit ISO 19115/19139 als Basis wurden folgende Erweiterungen realisiert:

1. bei den Datenprovidern Metadatenkataloge der lokal vorliegenden Daten + OAI-Server
2. zentrales Harvesten der lokalen Kataloge in einem Zentralkatalog nach OAI-PMH
3. Indizierung des Zentralkatalogs für schnelle Suche mit PanFMP (Beschreibung z.B. in [15])

beispielhafte Anwendung/Usecase: Standardisierte erweiterte Informationen über Ressourcen (hier Datensätze) im Grid austauschen.

Technologie: Monitoring & Discovery Service (MDS) [SDM+05] des Globus Toolkit 4 (GT4), OAI, OAI-PMH, PanFMP.

Schnittstellen: Metadatenkataloge - GUI

3.3 Workflow Schedulung Service (WSS)

Kurze Beschreibung: Im Rahmen von C3Grid wurde ein Workflowmanagement- und Scheduling-Dienst als eine zentrale Softwarekomponente entwickelt. Diese Komponente steuert einerseits die Interpretation und Ausführung von komplexen Klimaworkflows und versucht andererseits den Ablauf der Workflows für den Nutzer möglichst effizient zu gestalten. Im folgenden sind einige wichtige Aspekte der Softwarelösung aufgeführt und zudem wichtige architektonische Entscheidungen erläutert. Eine detaillierte Darstellung der Architektur und Funktionalität findet sich in [14].

Die in der Klimaforschung auftretenden Workflows beschreiben komplexe Prozessketten, die aus verschiedenen voneinander abhängigen atomaren Teilaufgaben (Tasks) zusammengesetzt sind. Diese Tasks sind u.a. Datenbeschaffung aus Klimadatenbanken, Datentransfer von Daten und die Analyse, Simulation der Modellberechnung auf Hochleistungsrechners. Abhängig vom Anwendungsfall oder Nutzerwunsch werden diese Tasks in vielfältiger Weise miteinander verbunden. Der C3Grid Workflow Scheduling Service (C3WSS) übernimmt die Verwaltung und effiziente Ausführung dieser Konstrukte. Dazu kooperiert der C3WSS mit dem Datenmanagementdienst des C3Grid und nutzt die Globus WS-GRAM-Schnittstellen der Rechenprovider. Insbesondere ist die Software selbst in das Web Service Resource Framework von Globus (WSRF) integriert. Für die Beschreibung der einzelnen Tasks wird auf die OFG-standardisierte Beschreibungssprache JSDL zurückgegriffen, während für die Beschreibung von Klimadatenbankoperationen sowie für die Darstellung von Abhängigkeiten proprietäre XML-basierte Sprachen benutzt werden. Die C3Grid-eigene Workflowbeschreibungssprache WSL erlaubt die Darstellung von Abhängigkeiten als gerichtete, azyklische Graphen. Aufgrund der fehlenden Anforderungen im C3Grid werden zyklische Bezüge in einem Workflow bisher nicht unterstützt.

Die Architektur des Dienstes folgt weitgehend dem Prinzip Service-orientierter Architekturen. So besteht der Workflow-Managementdienst selbst aus zwei weiteren Dienste: dem Jobmanagementdienst, der die Ausführung eines gesamten Workflows überwacht und steuert und dem Task-Managementdienst, der von ersterem genutzt wird, um einzelne Tasks auszuführen. Dieser Aufbau erlaubt eine Erweiterung der Taskfunktionalität (unterstützung zusätzlicher Taskarten) ohne daßdadurch die Workflowbearbeitung und das Scheduling auf höherer Ebene betroffen sind. Somit repräsentiert die Architektur des Dienstes direkt den Aufbau eines Workflow: einzelne Tasks werden von einzelnen Dienstinstanzen repräsentiert und durch den überliegenden Dienst koordiniert und miteinander verknüpft. Abbildung 4 zeigt schematisch den Aufbau des C3GridWSS.

Das Scheduling für jeden Workflow wird durch eine Strategie des Workflow-Managementdienstes ausgeführt. Diese ist als Modul auswechselbar und kann auf spezielle Bedürfnisse der Planung angepasst werden. Zur Zeit ist eine Strategie implementiert, die einerseits die Lokalität von Daten und andererseits die Auslastung und Verfügbarkeit von Ressourcen als Grundlage für die Verteilung atomarer Tasks nutzt. Sind Daten in der Nähe oder auf einer Ressource vorhanden wird versucht den Transportaufwand für die Eingabedaten eines Analyse- oder Simulationstasks durch dessen Verschiebung hin zu den Daten zu verringern. Dieses Vorgehen ist in den gewöhnlicherweise bewegten großen Datenmengen im C3Grid begründet. Da sich der Zustand des Grids (Auslastung,

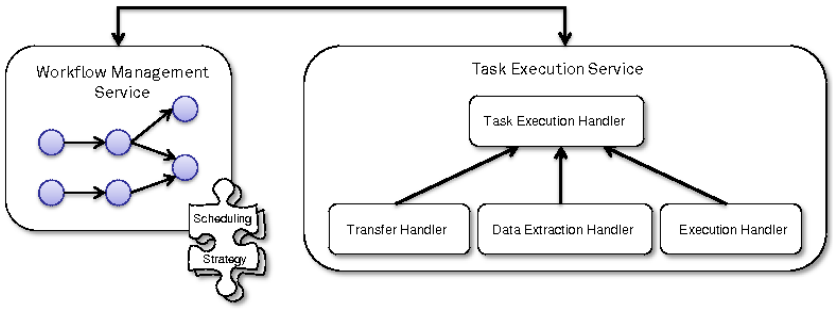

Abbildung 4: Schematische Darstellung der C3WSS-Architektur bestehend aus den Diensten zum Workflowmanagement und der Taskausführung.

Verfügbarkeit von Ressourcen) über die Zeit der Workflow-Ausführung ändern kann, wird vom Workflowmanagement kein fester Plan für die Ausfü hrung erstellt, sondern auf die Situation mit dynamischen Entscheidungen zur Ausführungszeit jedes Tasks im Workflow eingegangen.

Das Workflowmanagement wird in der Gesamtarchitektur des C3Grid nach oben durch das Portal verdeckt, das den Nutzern die Konfiguration von Workflows erlaubt und das Einreichen dieser übernimmt. Zudem arbeitet der C3WSS mit dem Datenmanagementdienst zusammen, der die Datenbezogenen Aktivitäten abstrahiert und ebenfalls über Web Service Schnittstellen zugreifbar macht.

Der Scheduler zeichnet sich durch enge Zusammenarbeit mit dem C3Grid Datenmanagement Service (GNDMS) aus (co-scheduling).

beispielhafte Anwendung/Usecase: Scheduling von Workflows unter Berücksichtigung der benötigten Datensätze und deren Bereitstellung.

Technologie: GT4, Java, JSDL.

Schnittstellen: Portal - Datenmanagementdienst

3.4 Datenmanagement Service (GNDMS)

Kurze Beschreibung: Das *Generation N Data Management System* (GNDMS) ist ein flexibles verteiltes Grid-Daten-Managementsystem. Zum Funktionsumfang gehören u.a. der Umgang mit logischen Pfadnamen, die Durchführung von Dateitransfers, Workspace-Management, und eine kontrollierbare, automatisierte Speicherressourcenverwaltung. GNDMS unterstützt Staging und Co-Scheduling und integriert sich als Menge von WSRF-Diensten für den Globus Container in die bestehende Grid-Infrastruktur.

GNDMS wurde im Rahmen von C3Grid entwickelt, ist aber flexibel und anpassbar und daher für den weiteren Einsatz in künftigen Grid-Projekten geeignet.

Das Datenmanagement in der Erdsystemforschung erfordert den Umgang mit hochvolumigen Datenmengen, heterogenen Datenbeständen und typischen Ausfall- und Fehlerszenarien verteilter Grid-

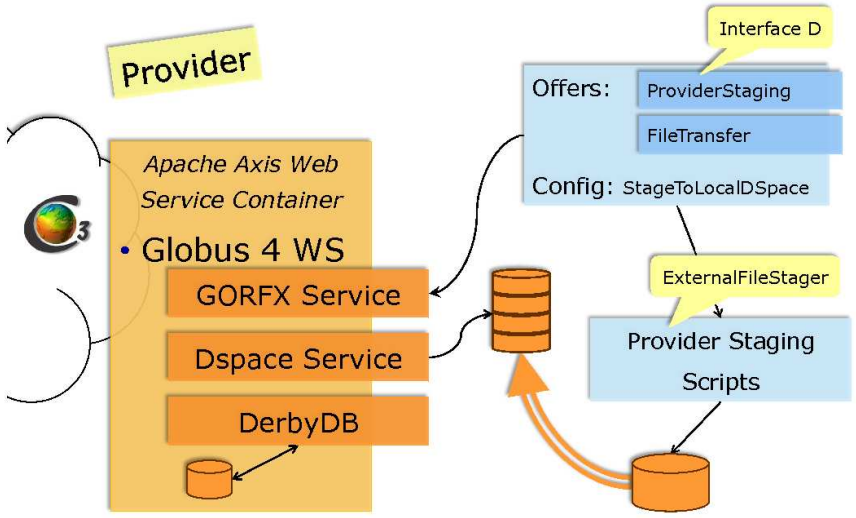

Abbildung 5: Komponenten des C3Grid bei den Datenprovidern.

Systeme. Hierfür wurde „Generation N Datenmanagementsystem (GNDMS)“ mit den folgenden Eigenschaften entwickelt:

- *Verteilt:* GNDMS wird auf allen Datenservern im deutschlandweiten C3Grid eingesetzt. Der Zugriff auf die einzelnen Datenprovider erfolgt über eine zentrale Master-Site, die für das C3Grid am ZIB betrieben wird. Verfügbare Datenprovider werden im verteilten und replizierten MDS-Verzeichnisdienst dynamisch katalogisiert.
- *Ausfallsicher:* GNDMS-Serverknoten speichern ihren vollständigen Zustand mit allen laufenden Datenanfragen in einer Datenbank und fahren nach einem Ausfall und Neustart mit der Abarbeitung ausstehender Aufträge fort.
- *Erweiterbar:* Durch eine Plugin-Architektur kann die GNDMS-Software mit geringem Aufwand um neue Methoden zur Datenbereitstellung erweitert werden. So ist es auch zukünftig möglich, neue Datenspeicher anzubinden.
- *Wartbar:* Um im Fehlerfall nachvollziehen zu können, welche Systemkomponente nicht ordnungsgemäßfunktioniert hat, können Log-Meldungen von GNDMS-Serverknoten in der Master-Site aggregiert und ausgewertet werden.
- *Effizient:* Die Master-Site wählt für eine Datenanfrage dynamisch den zuständigen Datenprovider so aus, daßUmfang und Dauer des Datentransports minimiert werden. Langfristig wird das System hierfür um intelligente Caching- und Replikatplatzierungsstrategien erweitert.
- *Flexibel:* Die GNDMS-Software aggregiert verschiedene Dienste, welche zusammen die Master-Site bilden. Es können aber auch einzelne Dienste auf anderen Grid-Sites installiert und mit anderen Rollen, z.B. der eines Datenanbieters, belegt werden.
- *Sicher:* GNDMS unterstützt die im Grid-Umfeld übliche GSI-Architektur, welche nur zertifizierten Nutzern den Zugriff auf Grid-Ressourcen erlaubt.

Zwei Komponenten der GNDMS-Software, die hier besonders hervorgehoben werden sollen, sind

der DSpace- und der GORFX-Dienst (siehe auch Abb. 5).

Schnittstellen: DSpace - GUI ; GORFX - GUI

DSpace (Data Space). Der DSpace-Dienst ermöglicht eine strukturierte, einheitliche Sicht auf alle Daten. Der Datenraum ist in Unterräume unterteilt, welchen jeweils Speicherplatz und eine Rolle zugeordnet sind, wie beispielsweise das „Stagen“ oder „Harvesten“. Einzelne Datensätze eines Unterraums sind in sog. Slices unterteilt. Diese können mit unterschiedlichen Zugriffsrechten versehen und verschiedenen Prozessen zugeordnet sein. Alle Slices werden nach Ablauf ihrer Lebenszeit automatisch gelöscht.

GORFX (Generic Offer Request Factory). Die zentrale Aufgabe des GORFX-Dienstes liegt in der Ausführung von Arbeitsaufträgen, wie z.B. das Bereitstellen oder Transferieren von Daten. Je nach Einsatzzweck von GORFX können einzelne solcher Funktionen aktiviert und konfiguriert werden, wodurch auch der GORFX-Dienst verschiedene Rollen annehmen kann.

Alle konfigurierten GORFX-Funktionen werden über ein Verhandlungsprotokoll (Offer-Request-Mechanismus), ein Kernbestandteil des GORFX-Dienstes, angeboten: Clients beauftragen den GORFX-Dienst mit der Erstellung eines Angebots für die Ausführung einer Datenmanagementaufgabe unter Berücksichtigung zeitlicher Rahmenbedingungen. Der GORFX-Dienst ermittelt das bestmögliche Angebot und legt es dem Client vor. Wenn dieser es innerhalb eines Zeitfensters akzeptiert, wird der Auftrag ausgeführt. Dieser Verhandlungsmechanismus ermöglicht die Erstellung von Ausführungsplänen für die Ausführung komplexer Workflows und die dynamische Verteilung von auszuführenden Aufträgen.

Derzeit umfasst der Java Code für das GNDMS ca. 36.000 Zeilen Code (ca. 63.000 Zeilen mit generierten Stubs).

beispielhafte Anwendung/Usecase:

- Co-scheduling für alle Datenmanagementaktivitäten
- Hot Failover und Failure Recovery für alle Datenmanagement-Aktivitäten basierend auf persistenter Speicherung des gesamten Systemzustands
- Interoperabel und grid-ready durch den Einsatz etablierter Standards wie Globus/WSRF, GSI und GridFTP Speichermanagement
- Hochgradig konfigurier- und erweiterbar durch generische Datenmanagement-Aktivitätsplugins und den Einsatz eines systemweiten Verzeichnisdienstes auf der Basis von MDS
- Handhabung temporärer Daten im Grid mit automatisiertem Life-Cycle
- Komponenten für das Systemmanagement: Laufzeit-Monitoring und verteiltes Logging

Technologie: GT4, Java, WSRF-Dienste, Apache Derby, Groovy, OpenJPA 2.0, Google Guice

3.5 Metadatensuchdienst

Kurze Beschreibung: Um verteilte Metadatenkataloge im Grid durchsuchbar zu machen wurde für die strukturierten Metadaten ein Suchdienst entwickelt, der eine Volltextsuche in den Metadaten erlaubt.

beispielhafte Anwendung/Usecase: Volltextsuche in strukturierten Metadaten und Textdateien.

Technologie: Apache Lucene mit Erweiterungen, OAI-PMH

4 Höhere Dienste in MediGrid

4.1 Überblick

In MediGRID werden die in Tabelle 3 aufgeführten Dienste als "höhere Dienste" im Sinne dieses Dokuments identifiziert.

Service	Technologie	allg. Beschreibung
Accounting-Service	CollAct, DGAS, Webservices, webbasierte GUI	Durchführung von fachlichem Accounting und Billing auf Ebene von Anwendungs-Services.
Ontology Service / Ontology Information Service	OGSA-DAI, WSRF	Zugriff auf unterschiedliche Ontologien bzw. deren Auswahl.

Tabelle 3: Identifizierte höhere Dienste in MediGrid [19].

Weitere zwischen der Middleware- und Anwendungsschicht befindliche Dienste werden eingesetzt, erfüllen allerdings nicht die Definition aus Kapitel 1.1, da sie beispielsweise nicht speziell für die Community MediGRID entwickelt bzw. eingesetzt werden. Hierzu zählen:

- Grid Workflow Execution Service (GWES)
- Portalsystem (Liferay, gPUT)
- Monitoring-Dienste (Ganglia, MDS, etc)
- Datenmanagement-Dienste (iRods, SRB etc.)
- D-Grid Resource Description Language (D-GRDL)
- etc.

Bei dem GWES handelt es sich um einen von mehreren Communities (Fraunhofer Enterprise Grids, Instant-Grid, MediGRID, BauVOGrid, TextGrid) weiterentwickelten Dienst, der aber nicht spezifisch für MediGRID entwickelt wurde. D-GRDL ist eine innerhalb des DGI-Projektes entwickelte Ressourcenbeschreibungssprache für beliebige Ressourcen, die das Scheduling und Monitoring von Ressourcen ermöglicht. Das Portalsystem basiert auf einem Standard-Portalsystem (derzeit GridSphere, in Zukunft LifeRay), welches für MediGRID angepasst wurde. Bei den eingesetzten Diensten zum Monitoring und Datenmanagement handelt es sich ebenfalls nicht um Community-spezifische Lösungen.

Middleware Als Middleware wird in MediGrid das Globus Toolkit GT4 verwendet.

4.2 Accounting-Service

Der Accounting-Service ("CollAct") ist für die Erfassung und Abrechnung der angefallenen Leistungen zwischen dem Anwendungsanbieter und dem Kunden zuständig. Die Hauptaufgabe besteht

darin die Nutzungsinformationen für eine spätere Rechnungsstellung zu sammeln. Diese Nutzungsinformationen, die Aussagen welche Nutzer eine Grid-Anwendung unter welchen Bedingungen genutzt hat, werden von den verschiedenen Grid-Anwendungen über einen Webservice an CollAct gesendet und dort gespeichert. Diese Nutzungsinformationen können anschließend von dem Anwendungsanbieter eingesehen werden und dienen als Basis für eine Rechnungserstellung.

Technologien: Apache Tomcat, JBoss, PostgreSQL, Webservices

Schnittstellen: Monitoring-Dienste, andere Accounting-Dienste (z.B. DGAS), Benutzeroberfläche

4.3 Ontology Service / Ontology Information Service

Der Ontology Service bietet mittels einer Erweiterung der OGSA-DAI-Schnittstelle den Zugriff auf unterschiedliche Ontologien (z.B. GenoOntology, National Cancer Institute, etc) an. Der Ontology Information Service ermöglicht das Auffinden von geeigneten Ontologien. Über einen Generic Ontology Client kann der Zugriff auf die beiden Dienste (Ontology Service, Ontology Information Service) aus einer Anwendung heraus erfolgen. Quelle: [20]

Technologien: GSA-DAI, Web Services Resource Framework (WSRF)

Schnittstellen: Anwendungen, Ontologien (z.B. GeneOntology, National Cancer Institute, etc)

5 Höhere Dienste in TextGrid

5.1 Überblick

Die in TextGrid entwickelten höheren Dienste zeichnen sich vor allem durch die Schwerpunkte auf Daten- und Metadatenverwaltung aus.

TextGrid entwickelt eine Virtuelle Forschungsumgebung für Philologen, Linguisten, Musikwissenschaftler und Kunsthistoriker. Das TextGridLab, ein intuitiv bedienbarer Einstiegspunkt, bietet Zugriff auf fachwissenschaftliche Werkzeuge, Services und Inhalte. Das TextGridRep, ein Grid-basiertes, fachwissenschaftliches Langzeitarchiv, garantiert die langfristige Verfügbarkeit und Zugänglichkeit der geisteswissenschaftlichen Forschungsdaten. Quelle: [8].

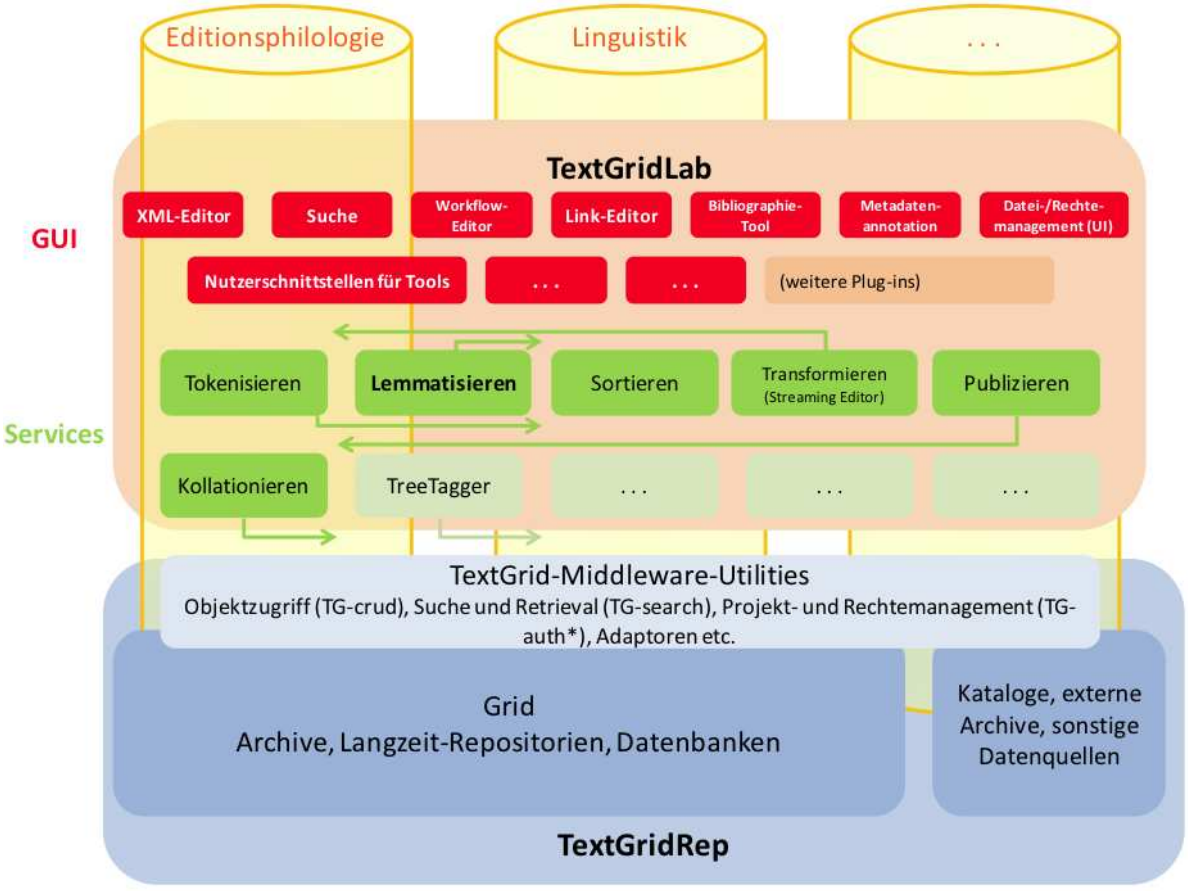

Abbildung 6: Architektur des TextGrids, TextGridLab und TextGridRep. Quelle:[25]

Die Architektur (siehe Abbildung 6) der Virtuellen Forschungsumgebung TextGrid besteht aus den beiden Hauptkomponenten Text-GridLab und TextGridRep.

Das TextGrid Laboratory ist ein Einstiegspunkt in die Virtuelle Forschungsumgebung und macht Tools und Services in einer auf Eclipse basierenden Arbeitsumgebung verfügbar. Beim TextGrid-Lab handelt es sich um eine Sammlung verschiedener Tools und Dienste für die Bearbeitung und Auswertung von textbasierten Daten in unterschiedlichen digitalen Archiven.

Das TextGrid Repository ist ein Grid-basiertes, fachwissenschaftliches Langzeitarchiv, in dem Daten gespeichert, archiviert, indiziert und semantisch annotiert werden. Quelle: [25]

Darüber hinaus bietet TextGrid eine offene, Web Service-basierte Infrastruktur, deren Komponenten sich auch außerhalb des Eclipse-basierten Frontends andressieren und nutzen lassen. Ebenso ist es möglich, neue Werkzeuge mit geringem Aufwand in diese Infrastruktur zu integrierenD.

Middleware In TextGrid wird als Middleware das Globus Toolkit GT4 eingesetzt. Desweiteren werden Metadaten-Middleware-Komponenten der Open Archives Community verwendet. Weiterhin kommen Eigenentwicklungen zur Verwaltung von Metadaten und zur Datenverwaltung zum Einsatz. Quelle: [2]

Middleware-Extensions aus Deliv. 2.1.1 In Deliverable 2.1.1 wird die filebasierte Daten/Metadatenverwaltung in Globus, die Metadaten-Suche, RDF-basierte Verwaltung von Objekt-Relationen sowie das Shibboleth/RBAC-basierte Rechtemanagement als Erweiterung der klassischen Middleware erwähnt. Auch diese soll hier genauer beschrieben werden. Quelle: [2]

5.2 Services im TextGridLab:

Services als Teil des TextGridLab sind die sogenannten "Streaming-Tools" (d.h. nicht interaktiv, sondern batch-processing), z.B. Tokenizer, Lemmatizer, Sortiertools, Streaming Editor, Web-Publishing Komponente. Es handelt sich hierbei um Web Services, welche über eine GUI-Komponente (Eclipse-Plugin) oder direkt über eine SOAP- oder REST-**Schnittstelle** angesprochen werden können. Sie können zu Workflows (Workflow-Manager s.u.) arrangiert werden und lassen sich ebenfalls mit geringem Aufwand in andere Applikationen integrieren. [26] Daneben existieren eine Reihe interaktiver Tools, wie z.B. XML-Editor, Text-Bild-Linkeditor u.a.m. [24]

5.3 Services im TextGridRep:

Bei den Services als Teil des TextGridRep handelt es sich zum einen um die sog. TextGrid Utilities (s.u.). Weiterhin ist ein Workflow Manager in die Middleware integriert. Dieser verwendet den Grid Workflow Execution Service (GWES) und ist sowohl vom Workflow Editor des TextGridLabs als auch über eine SOAP-Schnittstelle ansprechbar.

5.3.1 TexGrid Utilities

Es handelt sich bei den TexGrid Utilities um die Bausteine, die zwischen den einzelnen Grid-Diensten und der Basis-Middleware vermitteln. Namentlich sind dies TG-auth* (Rechtemanagement), TG-crud (Objektmanagement), TG-search (Suche) und TG-log (Logging). Auch diese Komponenten verfügen über eine SOAP- und/oder REST-Schnittstelle. Quelle: [27]

5.3.2 TG-Crud

TG-crud ist eine Komponente für das Daten-Management. Sie bietet die CRUD-Funktionalitäten (Create, Retrieve, Update und Delete) von TextGrid Objekten [28]. Diese stellen Container dar, die sowohl Daten als auch Metadaten beinhalten. Das TextGrid Metadaten-Scheme baut hierbei auf die Definitionen aus "Dublin Core" auf, erweitert diese jedoch und bietet die Möglichkeit, (Projekt-)spezifische Metadaten als Sub-Schema über einen eigenen Namensraum einzubetten.

Technologie: JavaGAT

5.3.3 TG-search

Hierbei handelt es sich um eine Komponente zur Suche und Analyse, wobei der Leistungsumfang über mittlere Such-Mechanismen hinausgeht. TG-Search bietet weitere Funktionalitäten, die spezifisch dem Bedarf der Literatur- und Sprachwissenschaften entsprechen. Unter anderem kann eine Suche über Objekt Meta-Daten, Beziehungen zu anderen Objekten, Voll-Textsuche sowie XQuery für XML-basierte Dateien sowie Kombinationen der o.a. Funkionalitäten durchgeführt werden.

Technologie: XML (TEI), eXist, Lucene, RDF

5.3.4 TG-auth*

Unterstützt die verschiedenen Prozesse durch ein Rollen-basiertes Benutzermanagement, welches auch Gruppen, Projekt Teams und deren Rechte verwaltet. TG-CRUD und TG-search agieren transparent mit TG-auth*, um ein adäquates Rechte-Managegement zu ermöglichen. Quelle:[27]

Technologie: Shibboleth, Open RBAC

5.3.5 TG-publish (in Entwicklung)

Ermöglicht das Einspielen von Massendaten in das TextGridRep sowie die Veröffentlichung von Forschungsdaten direkt aus dem TextGridLab. Bestandteil des Services ist die Metadaten-Validierung, Vergabe von Persistent Identifiern und ggf. von Metadaten zur Langzeitarchivierung. Eine auf dem Merritt Storage Interface [29] basierende Schnittstelle wird derzeit implementiert.

6 Höhere Dienste in HEPGrid

6.1 Überblick

Die in HEPGrid verwendeten oder entwickelten höheren Dienste sind überwiegend in das WLCG-Framework eingebettet und agieren als modulare Erweiterung der gLite-Middleware und dCache als Storage-Backend. In erster Linie handelt es sich um Dienste des Job-Managements, also Job-Monitoring, Usage-Monitoring, Job-Submission, Scheduling oder Visualisierung.

Tabelle 4:

Service	Technologie	Beschreibung
Visualisierung/ User Portal	GridSphere	
Ganga	Python, Kerberos, VOMS-Extensions	Frontend Job-Definition und -Management
RMOST	SQL, Gridsphere, R-GMA	Steuerungs- und Monitoring System für Grid-jobs der ATLAS-Experiment-Software
visualization tools	graphischer Daten-browser Kommandozeilen-werkzeug	
connection service		stellt Verbindung zwischen Grid-job und Visualisierungstool
name service		errichtet den Verbindungsdienst
AMON	Sensor fuer R-GMA	Job-Monitoring System aus dem HEP-Grid

Middleware-Extensions aus Deliv. 2.1.1 Im WissGrid Deliverable 2.1.1 wird Entwicklung des Datamanagment Frameworks dCache in Kooperation mit FermiLab aufgeführt.

Middleware Als Middleware in HEPGrid kommt gLite zum Einsatz.

6.2 Ganga

GANGA erlaubt die einfache Interaktion mit heterogenen Compute-Umgebungen, Konfiguration der Anwendungen und kohärente Organisation von Jobs. Die Funktionalitäten können über verschiedene Schnittstellen, den GANGA Public Interfaces (GPI), genutzt werden (siehe Abb. 7). Ein GANGA-Job setzt sich aus verschiedenen Komponenten zusammen. Alle Jobs müssen zumindest eine Anwendungs- und eine Backend-Komponente definieren, die die verwendete Software beziehungsweise das Prozessierungs-System (Hardware, OS) spezifizieren. Zusätzliche Komponenten können die I/O-Daten spezifizieren. CPU-intensive Jobs können zudem über eine Splitter-Komponenten verfügen, die es erlaubt, Jobs in unabhängige Sub-Jobs zu zerlegen, sowie die entsprechende Merger-Komponente zur Aggregation der Sub-Jobs.

Das GPI kann so konfiguriert werden, dass es den spezifischen Anforderungen und Kenntnissen des Anwenders entspricht. GANGA erlaubt es nicht, einmal abgeschickte Jobs zu modifizieren. Es ist allerdings einfach möglich, Jobs zu replizieren, und die Replikas zu verändern.

Alle Job-Objekte werden in einer Job-Repository-DB gespeichert, sowie alle assozierten Input-/Output-Daten in einem File-Workspace abegelegt. Sowohl das Repository als auch der Workspace können auf einem lokalen Dateisystem oder einem Remote-Server realisiert werden.

GANGA unterstützt die Handhabung von User-Credentials, wie Grid-Proxies, Proxies mit VOMS-Extensions, und Kerberos-Tokens um Zugriff auf AFS-Space[2] zu ermöglichen. Die Credentials können direkt über das GPI gemanaged werden. GANGA unterstützt ebenso verschiedene Security-Modelle, wie Kerberos-basierte lokale Sicherheits-Infrastukturen oder GSI, die parallel in einer GANGA-Session verwendet können.

Technologie: Python, Kerberos, VOMS-Extensions

Schnittstellen: GPI (Ganglia Public Interface), GUI, Python(Kommandozeile) - Job Plugins

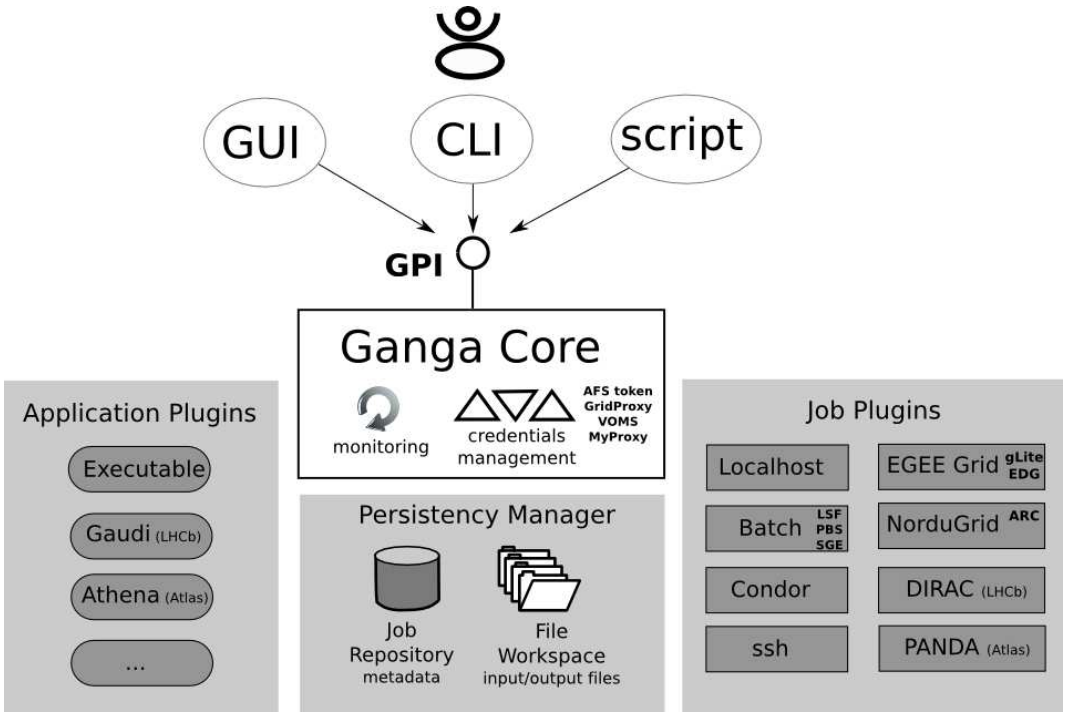

Abbildung 7: Ein Überblick über die Architektur von GANGA. Der Anwender interagiert mit der GANGA Schnittstelle (GANGA Public Interface, GPI) über das Graphical User Interface (GUI), das Command Line Interface in Python (CLIP), oder direkt mit Skripten. Plugins existieren für verschiedene Anwendung und Backends. Alle Jobs werden in dem Repository gespeichert. Quelle: [30]

[2]Das Andrew File System (AFS) gestattet den Export von Filesystemen über Firewalls. `http://en.wikipedia.org/wiki/Andrew_File_System`

6.3 RMOST

RMOST ist ein Online Steuerungs System, das speziell für Grid Jobs im Athena Framework entwickelt wurde, also der LHC-Software der ATLAS Collaboration. RMOST soll dazu dienen, die wesentlichen Aufgaben eines *computational steering systems* zu implementieren:

- Zugriff auf die laufende Grid-Anwendung für die Visualisierung temporärer Ergebnisse und Eingriff des Users in die Prozesse.
- Automatische Erstauswertung der Daten, automatische Fehlererkennung und Fehlerbehandlung.
- Management der Anwender-Aktionen, der Visualisierungsdaten und Steuerungsdaten.
- Der sichere Transfer von Daten und Befehlen zwischen dem Steuerungswerkzeug und dem Remote-Job.

Das RMOST Framework basiert auf einer Layer-Struktur (siehe Abb. 8), bestehend aus

- einem *Connection Layer*
- *einem Data Consistency Layer*
- *einem Data Processing Layer und*
- *einem Application Layer*

Dadurch ist es relativ einfach, einzelne Layer auszutauschen und damit RMOST für andere Applikation als die ATLAS Software und andere Middleware als gLite zu adaptieren.

Technologie: SQL, Gridsphere, R-GMA (siehe 6.4)

Schnittstellen: GUI, Daten, Befehle - R-GMA-Daten, Monitoring Skripte.

6.4 AMON

AMON [33] ist ein Job-Monitoring System aus dem HEP-Grid. Es ist in der Lage, detailierte Informationen über Nutzerjobs zu sammeln und diese an einen Grid-Monitoringdienst weiterzugeben. AMON ist primär in R-GMA integriert und kann daher als ein Sensor für R-GMA aufgefasst werden, der gegebenenfalls auch für andere Monitoringdienste erweitert werden könnte.

R-GMA (Relational Grid Monitoring Architecture [32]) ist eine in gLite integrierte, VO-spezifische, relationale und verteilbare virtuelle Datenbank, die es erlaubt die von AMON erhobenen Daten abzuspeicher und via Standard-SQL-Syntax wieder zu extrahieren.

AMON umfasst im Wesentlichen vier Komponenten (siehe Abb. 9):

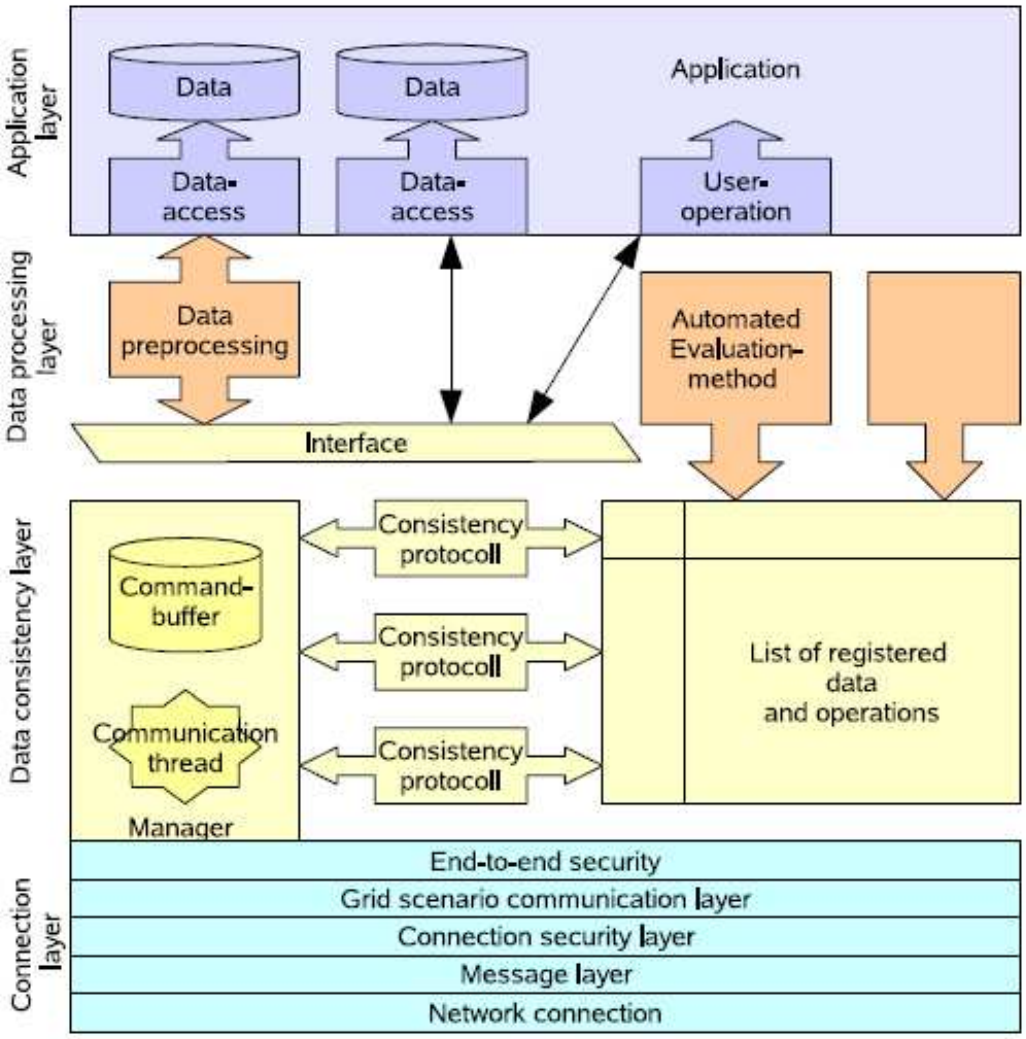

Abbildung 8: Die detaillierte Architektur der Implementation des RMOST Steering Frameworks. Quelle: [31].

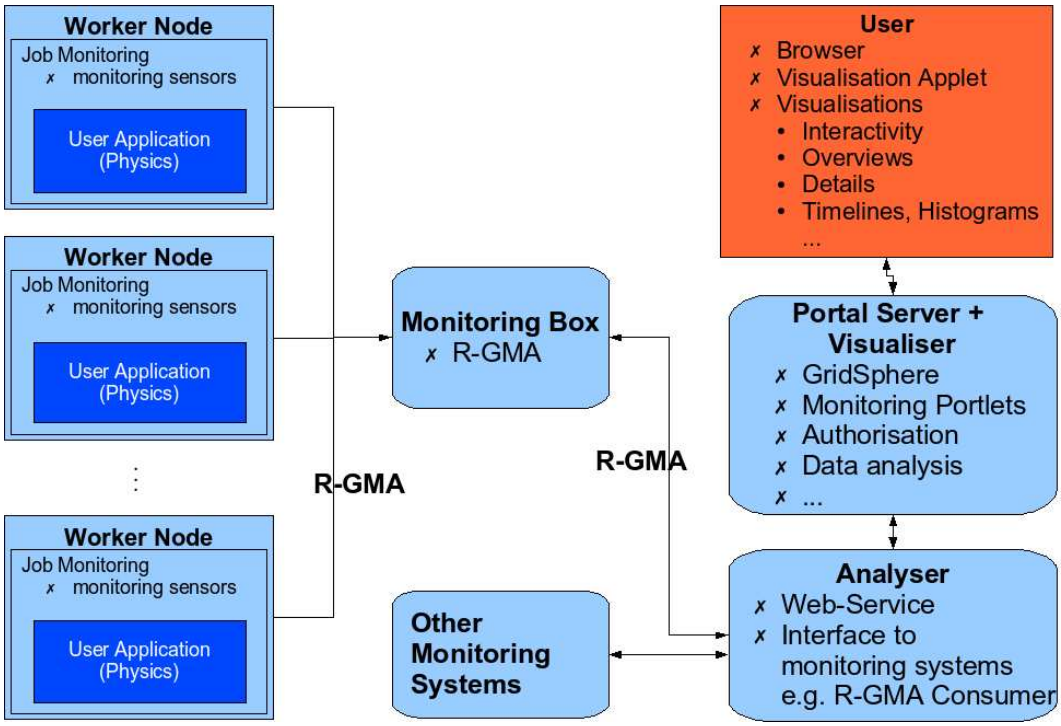

Abbildung 9: Die Architektur von AMON und die Einbettung in R-GMA. Quelle: [33].

- die lokale Datensammlung auf einzelen Grid-Knoten. Parallel zu einem Grid-Job wird ein Monitoring-Skript gestartet, das den Job überwacht. So genannte Sensoren können einzelne Aspekte des Jobs beobachten, wobei die Sensoren individuell definiert werden können, was die Implementierung neuer Sensoren vereinfacht.
- die Speicherung der Monitoring-Daten. Die Monitoring-Skripte schreibt die Daten in verschiedene Tabellen in R-GMA, verteilt auf verschiedenen Rechnern im Grid (*Monitoring Boxes).*
- *Sammlung der verteilten Informationen und erste Analyse der Monitoring-Daten Eine Anwendung, AMonAnalyser sammelt die verteilten Daten über einen WebService aus den einzelnen R-GMA Instanzen ab und bereitet die Daten für die Visualisierung auf.*
- *Aufbereitung und graphische Analyse der Monitoring-Daten. Die von ANomAnalyser aufbereiteten Daten werden über eine* **Schnittstelle** *(im HEP CG ein GridSphere Portal) dem Anwender zur Verfügung gestellt. Die Anwendung ANomVisualizer erlaubt dem Anwender die direkte Visualisierung der Daten.*

Literatur

[1] A. Rapp, C. Grimme, H. Enke, Community - Überblick / Report, WissGrid Deliverable 2.1.1, 23. Oktober 2009

[2] Deliverable 2.1 - Material Arbeitspaket 2: Blaupausen und Beratung Community - Überblick / Report, (Version 19.08.2009)

[3] AstroGrid-D Webseite, http://www.gac-grid.org

[4] Collaborative Climate Community Data and Processing Grid Webseite, http://www.c3grid.de/ C3-Grid

[5] MediGRID Webseite, http://www.medigrid.de/

[6] Das Hochenergiephysik Community Grid, Webseite bei D-Grid.de, http://www.d-grid.de/index.php?id=44

[7] HEP Community Grid - Projekt Status Webseite, http://documentation.hepcg.org/

[8] TextGrid Webseite, http://www.textgrid.de/

[9] Klaus Rieger, http://www.rzuser.uni-heidelberg.de/~tg5/, Juni 2010

[10] M. Hoegqvist, T. Roeblitz, A. Reinefeld, Stellaris: An RDF-based Information Service for AstroGrid-D, Conference-Paper, Mai 2007

[11] AstroGrid-D Information Service, Deliverable 2.4, Mai 2007

[12] Middleware-übergreifendes Monitoring: Evaluierung und Auswahl von Komponenten (AP 1.2), 29.2.2008

[13] H.Enke et al, AstroGrid-D: Grid Technology for the Astronomical Science, New Astronomy, in prep., 2010

[14] C. Grimme & A. Papaspyrou: Cooperative Negotiation and Scheduling of Scientific Workflows in the Collaborative Climate Community Data and Processing Grid. Future Generation Computer Systems, 25:301–307 (2008)

[15] Schindler, U., Bräuer, B., Diepenbroek, M.: Data Information Service based on Open Archives Initiative Protocols and Apache Lucene, GERMAN E-SCIENCE CONFERENCE, 2007. http://hdl.handle.net/10013/epic.26667

[16] Plantikowa, S., Petera, K., Hoegqvista, M., Grimme, C., Papaspyroub, A.: Generalizing the data management of three community grids, Future Generation Computer Systems, 25(3):281-289 (2009), http://dx.doi.org/10.1016/j.future.2008.05.001

[17] Tobias Langhammer, Florian Schintke, T2.1: Grid Information Service Architecture and Specification, März 2006

[18] Tobias Langhammer, Florian Schintke, T5.1: Grid Data Management Architecture and Specification, März 2006

[19] Julian Bart, Anwendungsintegration und Load-Balancing in MediGRID, D-Grid All-Hands Meeting, Workshop Portal Göttingen, 11. September 2007

[20] A Grid Middleware for Ontology Access, German e-Science Conference Baden Baden, 4. Mai 2007

[21] Andreas Hoheisel, Grid Workflow Executon Service (GWES 2.0) - Tutorial, Juni 2009

[22] Armin Wolf, Fraunhofer FIRST, Spezifikation der D-Grid-Ressourcenbeschreibungssprache D-GRDL und ihrer Nutzung im Grid-Computing, Nov 2007

[23] GWES Webseite bei Fraunhofer FIRST http://www.gridworkflow.org/kwfgrid/gwes/docs/index.html

[24] User's Manual TextGrid-Tools (25.10.2009) http://www.textgrid.de/fileadmin/TextGrid/reports/Report_2.3_final.pdf

[25] TextGrid Abschlussbericht (öffentliche Fassung, Version 1.0 vom 5. November 2009) http://www.textgrid.de/fileadmin/TextGrid/reports/TextGrid_Abschlussbericht_oeffentlich.pdf

[26] An Introduction to Development with TextGrid (3.8.2009) http://www.textgrid.de/fileadmin/TextGrid/reports/developers_tutorial_090803.pdf

[27] TextGrid Manual - Tool Development http://www.textgrid.de/fileadmin/TextGrid/reports/R3_5-manual-tools.pdf

[28] Installation eines Datengrid-Knotens http://www.textgrid.de/fileadmin/TextGrid/reports/TextGrid_Report_3_6-Datengrid-Knoten.pdf

[29] Merritt: An Emergent Micro-services Approach to Digital Curation Infrastructure https://wiki.ucop.edu/download/attachments/13860983/Merritt-latest.pdf

[30] J.T. Moscicki et al., Ganga: a tool for computational-task management and easy access to Grid resources, Computer Physics Communications 180(11): 2303-2316 (2009).

[31] D. Lorenz, RMOST Architecture, http://www.hep.physik.uni-siegen.de/grid/rmost/doc/Architecture.pdf

[32] Cooke, A. et al., The Relational Grid Monitoring Architecture: Mediating Information about the Grid. J. Grid Comput. 2(4): 323-339 (2004)

[33] Pfefferkorn, R-M. et. al.: AMON - a User-Friendly Job Monitoring for the Grid. In: Proceedings of the CoreGRID Symposium, Springer US (2007)

[34] D-Grid DGI Seite zu dCache http://dgi.d-grid.de/index.php?id=469