F&L-Grid: Eine generische Backup und Recovery Infrastruktur für das D-Grid

Markus Mathes, Steffen Heinzl, Roland Schwarzkopf, Bernd Freisleben
Fachbereich Mathematik und Informatik, Philipps-Universität Marburg
Hans-Meerwein-Str. 3, 35032 Marburg

{mathes,heinzl,rschwarzkopf,freisleb}
@informatik.uni.marburg.de

Abstract: Grid Computing wird oftmals zur Durchführung zeitintensiver Experimente, die eine enorme Menge an Daten produzieren, verwendet. Da existierende Backup und Recovery Lösungen basierend auf GridFTP oder RFT detaillierte technische Kenntnisse bezüglich Konfiguration und Nutzung erfordern, sind diese nicht unbedingt für alle Anwender geeignet. Viele Wissenschaftler bevorzugen eine möglichst einfache und bedienungsfreundliche Lösung, um ihre experimentellen Ergebnisse zu sichern.

Das F&L-Grid Projekt, welches die Entwicklung eines Grids für Forschung und Lehre beabsichtigt, ist ein Teil des D-Grid. Hauptziel von F&L-Grid ist der Entwurf und die Entwicklung einer generischen Backup und Recovery Infrastruktur für beliebige Grid-Umgebungen. Dieser Beitrag diskutiert den aktuellen Projektstatus von F&L-Grid und skizziert den Entwurf und die Implementierung der generischen Backup und Recovery Infrastruktur.

1 Einleitung

Grid Computing Umgebungen sind heterogene Sammlungen von Hard- und Software, die sich an verschiedenen Orten befinden und von verschiedenen Organisationen zur Verfügung gestellt werden. Die Hauptziele solcher Umgebungen sind die gemeinsame Nutzung von Ressourcen und Lösung von Problemen über die Grenzen von individuellen Institutionen hinweg [FKT01]. Um den Benutzern einen bequemen Zugriff auf Ressourcen über standardisierte Schnittstellen zu ermöglichen, verwendet man service-orientierte Grid Middleware basierend auf dem Web Service Resource Framework (WSRF) [OAS]. Beispiele für solche service-orientierte Grid Middleware sind das Globus Toolkit 4.x [Pet08] und Unicore/GS [Rom99]. Normalerweise unterstützt eine solche Middleware Laufzeitkomponenten, Ausführungs- und Informationsmanagement, Sicherheit und Datenhaltung. Unter Verwendung einer service-orientierten Grid Middleware werden Applikationen aus mehreren Grid Services komponiert. Ein Grid Service implementiert dabei einen kleinen Teil der gesamten Funktionalität. Große Applikationen werden in eine Vielzahl von Grid Services zerlegt, die dann zu neuen Applikationen komponiert werden können. Diese Vorgehensweise vermindert die redundante Implementierung von Funktionalität und erhöht gleichzeitig die Flexibilität.

Auf Grund ihrer Rechenleistung werden Grid Computing Umgebungen oftmals verwen-

det, um komplexe Experimente durchzuführen, z.B. innerhalb der Hochenergiephysik. Solche Experimente produzieren üblicherweise eine große Menge an Daten. Zum Zwecke der Datenhandhabung bieten viele service-orientierte Grid Middleware Systeme (funktional sehr eingeschränkte) Tools an. Beispielsweise bietet das Globus Toolkit GridFTP – eine Kommandozeilen-basierte FTP-Lösung für Grid-Umgebungen – und Reliable File Transfer (RFT) – einen Service-Wrapper für GridFTP. Diese mitgelieferten Tools werden häufig zum Backup und Recovery zweckentfremdet. Leider sind sie oftmals schwer zu bedienen und überfordern Wissenschaftler, die lediglich ihre Daten sichern wollen, ohne dabei die Interna der Grid Middleware zu kennen.

In diesem Beitrag wird eine generische Backup und Recovery Infrasktruktur präsentiert, die innerhalb des F&L-Grid Projektes entwickelt wurde. F&L-Grid ist Teil der D-Grid Initiative [NKG07] und wird von folgenden Projektpartnern getragen: T-Systems SfR und Karlsruhe Institute of Technology als Dienstanbieter, DFN-Verein als Anbieter der Netzwerkinfrastruktur und Philipps-Universität Marburg als Entwickler.

Im diesem Beitrag wird auf folgende Themen eingegangen:

- Die generellen Anforderungen an eine Backup und Recovery Infrastruktur für Grid Umgebungen werden identifiziert.
- Basierend auf den Anforderungen wird eine einfach zu verwendende Backup und Recovery Infrastruktur für beliebige Grid Umgebungen vorgestellt. Wissenschaftlern wird es ermöglicht, ihre Daten einfach und ohne das Eingreifen eines Administrators zu sichern und wiederherzustellen.
- Um Ausfallsicherheit zu garantieren, setzt die F&L-Grid Lösung auf einem kommerziellen Backup und Recovery Backend auf. Die Details der kommerziellen Lösung werden jedoch vor dem Benutzer verborgen.

Der Rest dieses Beitrags ist wie folgt organisiert: In Abschnitt 2 werden die Anforderungen an die F&L-Grid Backup und Recovery Lösung diskutiert und die entwickelte Architektur präsentiert. Abschnitt 3 präsentiert Implementierungsdetails. Weitere Backup und Recovery Lösungen für Grid-Umgebungen werden in Abschnitt 4 diskutiert. Abschnitt 5 fasst den gesamten Beitrag zusammen und gibt einen Ausblick auf zukünftige Entwicklungen.

2 Entwurf einer generischen Backup und Recovery Infrastruktur

Die Deutsche Grid-Initiative (D-Grid) [NKG07] wurde 2003 als Teil der nationalen e-Science Initiative des Bundesministeriums für Bildung und Forschung (BMBF) [BMB] gegründet. Ihr Ziel ist der Aufbau einer nachhaltigen Grid-Infrastruktur in Deutschland, um die Voraussetzungen für e-Science zu schaffen. D-Grid besteht aus mehreren Community-Projekten und dem D-Grid-Integrationsprojekt.

Das in diesem Beitrag vorgestellte F&L-Grid Projekt hat zum Ziel, ein generisches, service-orientiertes Grid für Forschung und Lehre aufzubauen. Die angebotenen Dienste werden

auf der Infrastruktur des DFN [DFN] bereitgestellt, die die Universitäten und Forschungseinrichtungen in Deutschland miteinander verbindet. In der aktuellen Projektphase wird ein Backup und Recovery Dienst entwickelt. Im Rahmen dieses Dienstes agieren einige Forschungseinrichtungen als Anbieter, andere als Benutzer, während das DFN die Rolle des Dienstvermittlers zwischen den beteiligten Einrichtungen übernimmt. Eine mögliche Erweiterung, die beispielsweise in einem Anschlussprojekt realisiert werden könnte, ist die Erweiterung um eine Archivierungsfunktion.

In den folgenden Abschnitten werden die Anforderungen von Anbietern und Benutzern dargestellt, pull- und push-basierte sowie Knoten- und Benutzer-basierte Ansätze zum Backup und Recovery verglichen, verschiedene Backup-Strategien erörtert und ein Überblick über die F&L-Grid Architektur gegeben.

2.1 Anforderungsanalyse

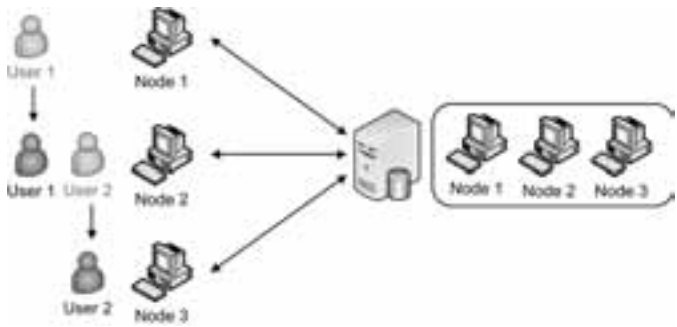
Die entscheidenden Anforderungen, die von F&L-Grid Anbietern und Benutzern gestellt wurden, sind:

- **Minimalinvasivität:** Der Backup und Recovery Dienst darf keine bereits existierenden betrieblichen Abläufe der Anbieter beeinflussen.
- **leichte Verwendbarkeit:** Es soll Wissenschaftlern ermöglicht werden, experimentelle Ergebnisse zu sichern oder wiederherzustellen, ohne dass Hilfe vom Administrator oder Help-Desk in Anspruch genommen werden muss, was bei kommerziellen Lösungen oft nötig ist.
- **einfache Installation:** Die notwendige Software soll sowohl auf Anbieter- als auch auf Benutzerseite einfach installiert werden können. Insbesondere Aktualisierungen der Client-Software sollen auf den Maschinen der Benutzer automatisch eingespielt werden.
- **Nachhaltigkeit:** Der entwickelte Backup und Recovery Dienst soll langfristig, also auch nach Abschluss des Projekts, vom DFN und den Anbietern angeboten werden können.
- **austauschbares Backend:** Eine Abhängigkeit des Dienst-Backends von einem speziellen kommerziellen System zur Backup und Recovery soll vermieden werden, so dass ein zukünftiger Austausch dieses Systems möglich ist.
- **Betriebssystemunabhängigkeit:** Es muss möglich sein, den Backup und Recovery Dienst auf verschiedenen Betriebssystemen zu benutzen, ohne dass verschiedene Versionen des Dienstes und der Client-Software gepflegt werden müssen.
- **Unterstützung verschiedener Schnittstellen:** Der Backup und Recovery Dienst soll sowohl innerhalb als auch außerhalb von Grids verwendbar sein.

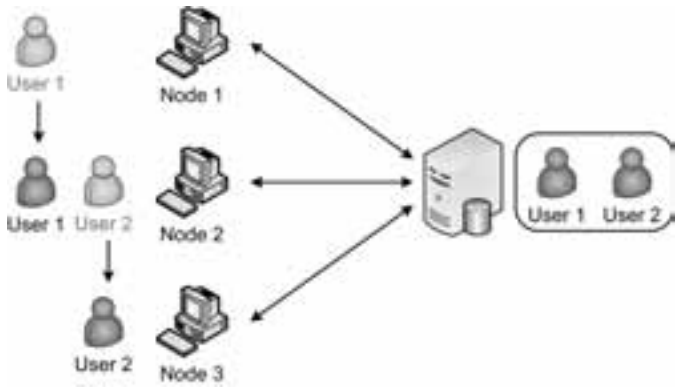
2.2 Pull- vs. Push-basierte Backup und Recovery Ansätze

Es gibt zwei Möglichkeiten, ein Backup durchzuführen: pull- und push-basiert. Beim *pull-basierten* Ansatz wird das Backup von der Server-Seite der Backup-Lösung gestartet. Dabei kommt meist ein voreingestellter Zeitplan zum Einsatz, der die Backups zu Zeiten mit niedriger Auslastung durchführt. Beim *push-basierten* Ansatz wird das Backup, z.B. nach dem Abschluss eines Experiments, vom Benutzer manuell angestoßen.

Da eine push-basierte Client-Software ohne Eingriff eines Administrators installiert und aktualisiert werden kann, wurde dieser Ansatz von den Projektpartnern gewählt.



(a) Knoten-basiertes Backup und Recovery.



(b) Benutzer-basiertes Backup and Recovery.

Abbildung 1: Beispiel eines Knoten- und Benutzer-basierten Backup und Recovery Ansatzes.

2.3 Knoten- vs. Benutzer-basierte Backup und Recovery Ansätze

Viele kommerzielle Lösungen unterstützen nur einen *Knoten-basierten* Ansatz für Backup und Recovery, weil sich die Dateisysteme auf unterschiedlichen Knoten erheblich unterscheiden können. Allerdings hat der Knoten-basierte Ansatz einen wesentlichen Nachteil:

ein Benutzer, der auf mehreren Knoten arbeitet, kann jeweils nur die Daten des gerade von ihm benutzten Knotens wiederherstellen, aber nicht alle seine Daten. Um dieses Problem zu lösen, muss ein *Benutzer-basierter* Ansatz gewählt werden, der eine Knoten-übergreifende Wiederherstellung von Daten ermöglicht.

Abbildung 1(a) zeigt einen Knoten-basierten Backup und Recovery Ansatz. Benutzer 1 und Benutzer 2 können ihre Daten nicht wiederherstellen, nachdem sie von Knoten 1 zu Knoten 2 bzw. von Knoten 2 zu Knoten 3 gewechselt sind. Mit dem Benutzer-basierten Ansatz wird eine Wiederherstellung von Daten auch nach einem Wechsel des Knotens möglich, wie Abbildung 1(b) zeigt.

Im Rahmen des F&L-Grid Projekts wird der Benutzer-basierte Backup und Recovery Ansatz realisiert.

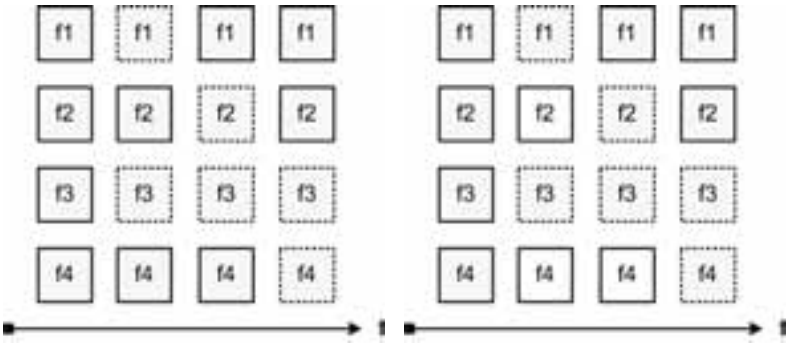
2.4 Inkrementelles, differentielles und Voll-Backup

Man unterscheidet zwischen 3 Backup-Strategien: inkrementelles, differentielles und vollständiges Backup. Ein *Voll-Backup* enthält alle ausgewählten Dateien. Ein *differentielles Backup* enthält alle Dateien, die seit dem letzten Voll-Backup geändert wurden. Ein *inkrementelles Backup* enthält nur die Dateien, die seit dem letzten inkrementellen Backup oder Voll-Backup geändert wurden.

In den Abbildungen 2(a), (b) und (c) werden Beispiele für ein Voll-Backup sowie ein inkrementelles und differentielles Backup von vier Dateien gegeben. Die Rechtecke stellen Dateien dar. Gestrichelte Linien zeigen eine Änderung seit dem letzten Backup an, durchgehende Linien stehen für unveränderte Dateien. Eine (gelbe) Füllung eines Rechtecks zeigt an, dass die Datei im jeweiligen Backup enthalten ist, wohingegen ungefüllte Rechtecke Dateien anzeigen, die nicht enthalten sind.

Der größte Vorteil der inkrementellen Backup-Strategie ist eine kürzere Backup-Dauer und geringerer Speicherverbrauch, weil nur die Änderungen seit dem letzten Backup enthalten sind. Um die Dateien wiederherzustellen, müssen das letzte Voll-Backup und alle danach durchgeführten inkrementellen Backups wiederhergestellt werden, was zu einer langen Recovery-Dauer führt. Die Verwendung der differentiellen Backup-Strategie verkürzt die Recovery-Dauer, weil nach dem Voll-Backup nur das letzte differentielle Backup wiederhergestellt werden muss. Dieser Vorteil geht jedoch zu Lasten der Backup-Dauer und des Speicherverbrauchs.

Im Rahmen des F&L-Grid Projekts wurde von den Partnern die Voll-Backup Strategie aus der Sicht des Fat-Clients (siehe Abschnitt 2.6) ausgewählt. Das ist nötig, weil der Fat-Client unter anderem als Zwischenspeicher fungiert und die Benutzerdaten nur temporär speichert, was differentielle oder inkrementelle Backups bei verschiedenen Backup und Recovery Lösungen ausschließt. Aus Sicht des Benutzers wird eine inkrementelle Backup-Strategie umgesetzt, da immer nur die seit dem letzten Backup modifizierten Daten zum Fat-Client übertragen werden.



(a) Voll-Backup. (b) Differentielles Backup.

(c) Inkrementelles Backup.

Abbildung 2: Beispiele für die unterschiedlichen Backup-Strategien (gestrichelte Rechtecke sind nach dem letzten Backup modifizierte Dateien, eingefärbte Rechtecke sind im aktuellen Backup enthaltene Dateien).

2.5 Metadaten

Die Metadaten zu den Dateien (z.B. Zugriffsrechte, Zeitstempel, etc.) unterscheiden sich auf den verschiedenen Dateisystemen (z.B. NTFS, ReiserFS, ZFS) und Betriebssystemen (z.B. Windows, Linux, MacOS). Folglich ist eine einheitliche Repräsentation dieser Metadaten ein schwieriges Unterfangen.

In F&L-Grid wird eine Schnittmenge von Metadaten verwendet (z.B. Datei-/Verzeichnisname, Zeitpunkt der Erzeugung und letzten Änderung, Dateigröße) die auf möglichst viele Betriebssysteme übertragbar ist. Alle relevanten Metadaten werden in einer Datenbank auf dem Fat-Client gespeichert (siehe Abschnitt 2.6).

2.6 Architekturskizze

Die Architekturskizze in Abbildung 3 spiegelt die identifizierten Anforderungen wider.

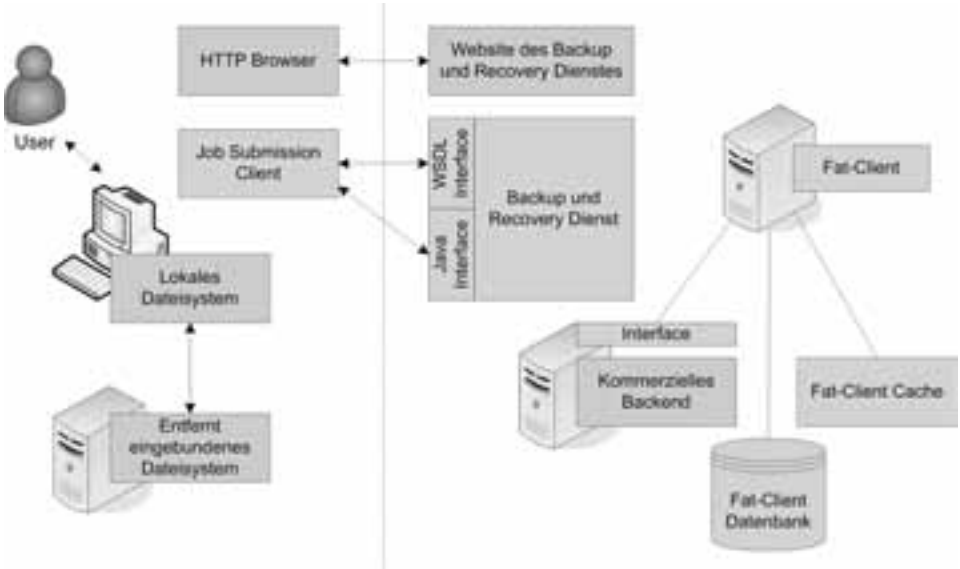


Abbildung 3: Architektur des F&L-Grid Backup und Recovery Dienstes.

Auf der Benutzer-Seite wird der sogenannte *Job Submission Client* benötigt, um auf die Backup und Recovery Infrastruktur zuzugreifen. Diese Software wird auf einer Website angeboten und kann mit dem Browser heruntergeladen werden. Damit kann der Benutzer jedes lokale oder per Netzwerk eingebundene Laufwerk sichern.

Die wichtigste Komponente auf der Anbieter-Seite ist der sogenannte *Fat-Client*. Er ermöglicht einen einfachen Zugriff auf die Backup und Recovery Infrastruktur und versteckt dabei viele Details vor dem Benutzer. Dabei übernimmt er die Kommunikation mit dem Backup und Recovery Backend, verwaltet zum Backup gehörige Daten in einer speziellen Datenbank und pflegt seinen Cache. Als Backup und Recovery Backend kann eine beliebige kommerzielle Lösung eingesetzt werden, da die Schnittstelle beliebig austauschbar ist. In der Datenbank werden die Metadaten und abrechnungsrelevanten Informationen, z.B. wie viele Backups ein bestimmter User ausgeführt hat, gespeichert. Der Cache wird benutzt, um die zu einem Backup oder Recovery Vorgang gehörenden Daten bis zu dessen Abschluss vorzuhalten. Das kommerzielle Backend liest beim Backup aus diesem Cache und schreibt beim Recovery in ihn. Diese Abläufe werden vom Backup und Recovery Dienst koordiniert.

2.7 Backup

Ein Backup-Vorgang besteht aus sechs aufeinander folgenden Schritten, die in Abbildung 4 dargestellt sind.

Abbildung 4: Bearbeitung eines Backup-Vorgangs.

1. **Authentication/Authorization:** Der Benutzer, der einen Backup-Vorgang anstoßen will, wird identifiziert, und der Zugriff auf das System wird ihm gestattet.

2. **Scan:** Die Metadaten von allen für das Backup ausgewählten Dateien/Verzeichnissen werden eingelesen und zu einer Liste zusammengefasst.

3. **Intersection:** Um unnötiges Kopieren von Daten zwischen dem Knoten des Benutzers und dem Fat-Client zu vermeiden, werden in diesem Schritt alle Dateien/Verzeichnisse von der Liste entfernt, die sich seit dem letzten Backup nicht mehr geändert haben. Dazu werden die Informationen über bereits gesicherte Dateien aus der Datenbank verwendet.

4. **Copy:** Alle noch auf der Liste stehenden Dateien/Verzeichnisse werden vom Knoten des Benutzers über eine gesicherte Verbindung in den Cache des Fat-Client kopiert.

5. **Backup:** Die kommerzielle Backup und Recovery Lösung wird verwendet, um die Dateien im Cache des Fat-Client zu sichern. Danach werden die Informationen über gesicherte Dateien/Verzeichnisse in der Datenbank aktualisiert.

6. **Cleanup:** Im letzten Schritt werden die Dateien dieses Backup-Vorgangs wieder aus dem Cache gelöscht, um Platz für folgende Backup und Recovery Vorgänge zu schaffen.

3 Implementierung

Dieser Abschnitt gibt einen Überblick der ausgewählten Technologien zur Implementierung des Backup und Recovery Dienstes, erklärt, wie auf den Backup und Recovery Dienst zugegriffen werden kann und beschreibt die Verarbeitung eines Backup Jobs im Detail.

3.1 Auswahl geeigneter Technologien

Basierend auf den identifizierten Anforderungen wurden die folgenden Technologien zur Implementierung des Backup und Recovery Dienstes ausgewählt.

Um *Minimalinvasivität* zu garantieren, wurde Java Web Start für die Implementierung des Job Submission Client gewählt. Eine Java Runtime Environment (JRE) [Mic] ist heute auf fast allen Computern verfügbar, so dass die Installation zusätzlicher Software vermieden werden kann. Außerdem sichert die Verwendung von Java die *Unabhängigkeit vom verwendeten Betriebssystem*. Mit Java Web Start wird eine neue Version der Software bezogen, sobald diese zur Verfügung steht. Außerdem erlaubt Java Web Start die Signatur der Software zu einem vertrauenswürdigen Zertifikatsherausgeber zurück zu verfolgen. Diese Funktionalitäten ermöglichen einen *einfachen Software Roll-out*.

Durch die Verwendung generischer Java Interfaces wird es möglich, verschiedene kommerzielle Backup und Recovery Lösungen als Backend einzusetzen (z.B. IBM Tivoli Storage Manager, EMC NetWorker), was den *einfachen Austausch des Backup und Recovery Backend* ermöglicht. Eine klar strukturierte und übersichtliche GUI ermöglicht dem Benutzer eine *einfache Bedienung*. Um einen Zugriff auf den Backup und Recovery Dienst über verschiedene Schnittstellen zu ermöglichen, wurde eine generische Schnittstelle definiert, aus der beliebige konkrete Schnittstellen abgeleitet werden können, z.B. eine Beschreibung in der Web Service Description Language (WSDL) [W3C06]. Um die *Nachhaltigkeit* des angebotenen Dienstes zu garantieren, hat der DFN-Verein ein Geschäftsmodell entwickelt, das die Rechte und Pflichten auf Nutzer- und Anbieterseite regelt.

In einer Umfrage, welche durch den ZKI Arbeitskreis Netzwerkdienste [ZKI] durchgeführt wurde, wurden 41 wissenschaftliche Einrichtungen in Deutschland nach deren Backup und Recovery Software befragt. Da 66% dieser Institutionen bereits IBM Tivoli Storage Manager (TSM) verwenden, basiert die prototypische Implementierung des Backup und Recovery Dienstes ebenfalls auf IBM TSM als Backend. Da IBM TSM lediglich Knoten-basiertes Backup/Recovery ermöglicht, bietet unsere Implementierung zusätzliche Funktionen für Benutzer-basiertes Backup/Recovery. Alle Informationen für ein Benutzer-basiertes Backup/Recovery sowie Accounting und Billing werden in der Fat-Client Datenbank gespeichert. Folglich bietet die Fat-Client Datenbank einen globalen Blick auf den gesamten Backup und Recovery Dienst.

3.2 Zugriff auf den Backup und Recovery Dienst

Ein Benutzer des Backup und Recovery Dienstes benötigt den Job Submission Client, um auf den Dienst zugreifen zu können, d.h. um einen Backup oder Recovery Job abzusetzen. Hierzu lädt der Benutzer den Job Submission Client in Form einer Java Web Start Anwendung von einer Website herunter. Der Benutzer muss den Job Submission Client nur einmal manuell herunterladen. Anschließend werden neue Version automatisch durch die Java Web Start Technologie heruntergeladen.

Um die Software herunterladen zu können, muss sich der Benutzer gegenüber seiner Hei-

matorganisation via Shibboleth [SJWA06] authentifizieren (*Authentication/Authorization Phase*). Beispielsweise kann sich der Benutzer mittels Benutzername und Passwort beim Identity Provider seiner Heimateinrichtung authentifizieren. Shibboleth wurde verwendet, da es sich sehr gut in das existierende Service-Portfolio des DFN, welches ebenfalls Shibboleth benutzt, integriert.

3.3 Absetzen eines Backup-Jobs

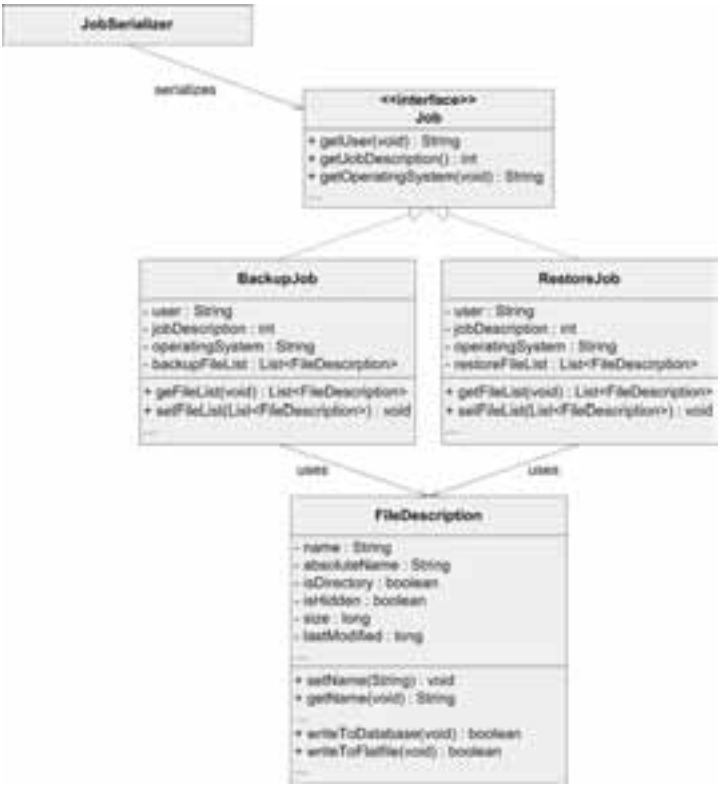


Abbildung 5: Klassenhierarchie `BackupJob` und `RecoveryJob`.

Ein Backup-Job wird durch die Auswahl aller relevanten Dateien und Verzeichnisse mit Hilfe des Job Submission Client definiert (*Scan Phase*). Der Backup-Job beschreibt die Dateien und Verzeichnisse durch deren Namen, den Hostnamen, von dem das Backup erfolgt, den Benutzernamen, den Zeitstempel der letzten Änderung und die Zugriffsrechte. Dieses `Job`-Objekt kann dann zum Backup und Recovery Dienst in Form eines Grid Service Aufrufs oder als ein serialisiertes Java-Objekt gesendet werden, je nach verwendetem Interface. Der Dienst entfernt alle Dateien/Verzeichnisse von der Liste, die seit dem letzten Backup nicht modifiziert wurden (*Intersection Phase*). Dies kann durch eine Anfrage bei

der Fat-Client Datenbank mittels JDBC ermittelt werden. Die modifizierte Liste wird an den Job Submission Client zurückgesendet, der die Daten anschließend mit Hilfe des Secure Copy Protocol (SCP) über eine Secure Shell (SSH) Sitzung an den Fat-Client sendet (*Copy Phase*). Der Prototyp verwendet die Java Secure Channel (JSch) [JSc] Bibliothek, welche eine Implementierung von SSH und SCP anbietet. Nachdem die Daten vollständig übertragen wurden, wird ein TSM Kommandozeilen-Client auf Seite des Fat-Client gestartet, um die Daten ins Backend zu übertragen (*Backup Phase*). Sobald das Backup erfolgreich durchgeführt wurde, werden Informationen zum Backup-Job in der Fat-Client Datenbank hinterlegt und die kopierten Daten werden aus dem Fat-Client Cache entfernt (*Cleanup Phase*). Die Zugriffsrechte werden für jede Datei als String hinterlegt. Derzeit werden UNIX Zugriffsrechte und Windows NTFS Zugriffsrechte mit Hilfe des Windows-Tools *cacls* (change access control lists) unterstützt.

Die Klassenhierarchie von Backup und Recovery Jobs wird in Abbildung 5 gezeigt. Die wesentlichen Eigenschaften eines `BackupJob` und eines `RecoveryJob` sind im Interface `Job` gekapselt. Beide spezialisierten `Job`-Klassen benutzen eine `FileDescription`, um relevante Dateien und Verzeichnisse zu beschreiben. Der `JobSerializer` wird benutzt, um einen Grid Service Aufruf oder ein serialisiertes Java-Objekt zu erzeugen.

4 Verwandte Arbeiten

Derzeit existieren nach unserem Wissen keine generischen und einfach verwendbaren Backup und Recovery Lösungen für Grid-Umgebungen. Folglich beschränken sich die verwandten Arbeiten auf Standardtechnologien für Datentransport in Grid-Umgebungen: GridFTP, RFT, RLS und OGSA-DAI.

Aktuelle service-orientierte Grid Middleware bietet von Haus aus Funktionalität zur Verwaltung von Daten. Basierend auf dieser Funktionalität wurden in einigen Forschungsprojekten oftmals proprietäre Backup und Recovery Lösungen entwickelt. Das Globus Toolkit 4.x [Pet08] beispielsweise bietet Funktionen zur Datenübertragung (GridFTP und Reliable File Transfer (RFT)) und Datenreplikation (Replica Location Service (RLS)). GridFTP ist eine für Grid-Umgebungen optimierte Version des weitverbreiteten File Transfer Protocol (FTP) und bietet eine effiziente, sichere und robuste Übertragung von Daten. Das Globus Toolkit beinhaltet einen GridFTP Server (`globus-gridftp-server`) und einen GridFTP Kommandozeilen-Client (`globus-url-copy`), um Daten bereitzustellen und zu übertragen. RFT bietet eine service-orientierte Schnittstelle zu GridFTP und ermöglicht zusätzlich das Scheduling von Datenübertragungsjobs. Die Replikation von Daten innerhalb eines Grids verbessert die Zugriffseffizienz. Um Replikate zu verwalten, bietet das Globus Toolkit RLS – eine einfache Registry für selbige. Die Verwendung von Middleware-spezifischer Funktionalität zur Datenverwaltung führt zu zwei Hauptproblemen:

- Die Backup und Recovery Lösung kann nur mit Aufwand wiederverwendet werden, da sie an die Middleware gebunden ist.

- Detailwissen über die Interna der Middleware ist erforderlich. Dies überfordert jedoch oftmals einfache Benutzer und macht das Eingreifen eines Administrators notwendig.

OGSA-DAI [ACHH+07] bietet einen Dienst, um auf Daten zuzugreifen, die aus verschiedenen Quellen stammen, z.B. Datenbanken oder flache Dateien. Außerdem bietet OGSA-DAI Funktionen, um Daten abzufragen, zu transformieren und auf verschiedene Weise auszuliefern. OGSA-DAI kann zwar verwendet werden, um Kopien von Daten anzulegen, jedoch ist die Durchführung zuverlässiger Backups damit nur schwierig möglich.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde eine generische Infrastruktur für Backup und Recovery in Grid-Umgebungen präsentiert.

Im F&L-Grid Projekt als Teil der D-Grid Initiative wurden zunächst die Anforderungen für eine solche, generische Backup und Recovery Infrastruktur identifiziert, und anschließend wurde eine geeignete Architektur entworfen. Der Backup und Recovery Dienst ist einfach zu benutzen und kann in beliebigen service-orientierten Grid-Umgebungen eingesetzt werden. Wissenschaftlern wird es ermöglicht, ihre Daten schnell, einfach und ohne das Eingreifen eines Administrator zu sichern und wiederherzustellen. Als Backend kann eine beliebige, kommerzielle Backup und Recovery Lösung eingesetzt werden, deren Details vor dem Anwender versteckt werden.

Zukünftig sollen weitere Metadaten für Dateien und Verzeichnisse unterstützt werden, z.B. die Vererbung von Zugriffsrechten. Außerdem wird eine Accounting und Billing Komponente entwickelt werden, welche Informationen aus der Fat-Client Datenbank verwendet, um automatisch Backup/Recovery Jobs abrechnen zu können. Ebenfalls von Bedeutung ist die Durchführung von Performance- und Skalierbarkeitsuntersuchungen vor dem Produktiveinsatz.

Danksagung

Diese Arbeit wird durch das Bundesministerium für Bildung und Forschung (BMBF) im Rahmen der D-Grid Initiative (F&L-Grid) finanziell unterstützt (`http://www.bmbf.de/`).

Literatur

[ACHH+07] M. Antonioletti, N.P. Chue Hong, A.C. Hume, M. Jackson, K. Karasavvas, A. Krause, J.M. Schopf, M.P. Atkinson, B. Dobrzelecki, M. Illingworth, N. McDonnell, M. Parsons, and E. Theocharopoulos. OGSA-DAI 3.0 The Whats and the Whys. In *Proceedings of the UK e-Science All Hands Meeting*, 2007.

[BMB] German Federal Ministry of Education and Research (BMBF). `http://www.bmbf.de/`.

[DFN] German Research Network (DFN). `http://www.dfn.de/`.

[FKT01] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, 15:200–222, 2001.

[JSc] JSch – Java Secure Channel (Project Homepage). `http://www.jcraft.com/jsch/`.

[Mic] Sun Microsystems. Java 2 Platform, Standard Edition. `http://java.sun.com/j2se/1.4.2/download.html`.

[NKG07] H. Neuroth, M. Kerzel, and W. Gentzsch. *German Grid Initiative (D-Grid)*. Niedersächsische Staats- und Universitätsbibliothek, 2007.

[OAS] OASIS. Web Services Resource Framework (WSRF). `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf`.

[Pet08] D. Petcu. A Comprehensive Development Guide for the Globus Toolkit. *Distributed Systems Online*, 9:4–6, 2008.

[Rom99] M. Romberg. The UNICORE Architecture: Seamless Access to Distributed Resources. In *Proceedings of the* 8^{th} *IEEE International Symposium on High Performance Distributed Computing (HPDC)*, pages 287–293. IEEE Computer Society Press, 1999.

[SJWA06] R.O. Sinnott, J. Jiang, J. Watt, and O. Ajayi. Shibboleth-based Access to and Usage of Grid Resources. In *Proceedings of the* 7^{th} *IEEE/ACM International Conference on Grid Computing*, pages 136–143. IEEE Computer Society Press, 2006.

[W3C06] W3C. Web Services Description Language (WSDL) 2.0, June 2006. `http://www.w3.org/TR/wsdl20/`.

[ZKI] Zentren für Kommunikation und Informationsverarbeitung in Lehre und Forschung e.V. (ZKI), Arbeitskreis Netzdienste. `http://www.zki.de/ak_nd/`.