Erfahrungen mit Grid-Technologien im Projekt WISENT

Arbeitspaket 1.2

Sönke Brummerloh, Jan Ploski, Guido Scherp
OFFIS e. V.
Escherweg2
26121 Oldenburg

Telefon: +49 441 9722-0
Telefax: +49 441 9722-102
E-Mail: {Soenke.Brummerloh, Jan.Ploski, Guido.Scherp}@offis.de
Internet: http://www.offis.de

Johannes Hurka
AG Energiemeteorologie - Energie- und Halbleiterforschung Institut für Physik
Universität Oldenburg
26111 Oldenburg

Telefon: +49 441 798 3927
Telefax: +49 441 798 3326
E-Mail: johannes.hurka@uni-oldenburg.de
Internet: http://www.energiemeteorologie.de

Gerhard Gesell
Deutsches Zentrum für Luft- und Raumfahrt DLR, Deutsches
Fernerkundungsdatenzentrum DFD
Münchner Straße 20
82234 Weßling

Telefon: +49 8153 28 1322
Telefax: +49 8153 28 1363
E-Mail: gerhard.gesell@dlr.de
Internet: www.caf.dlr.de

Sina Lohmann, Luca Bugliaro
DLR Oberpfaffenhofen
Institut für Physik der Atmosphäre
82234 Weßling

Telefon: +49 8153 282582
Telefax: +49 8153 281841
E-Mail: luca.bugliaro@dlr.de
Internet: http://www.dlr.de/pa/

1.Einleitung

Im Rahmen des Arbeitspakets 1.1 wurde eine Grid-basierte Infrastruktur aufgebaut, die im Laufe des Projekts in Arbeitspaket 1.2 genutzt und auch erweitert wurde. Aus Software-Sicht werden dazu Batch-Systeme und Grid-Middleware für den Zugang zu Rechenressourcen eingesetzt, sowie verschiedene Technologien zum Datenmanagement. Die Erfahrungen, die im Rahmen von WISENT bzgl. des Einsatzes von Grid-Technologien gemacht werden, sind in diesem Dokument festgehalten worden. Dabei stammen die Erfahrungen sowohl aus dem Arbeitspaket 1, welches im Projekt eine übergeordnete Rolle gespielt hat, als auch aus den einzelnen fachlichen Arbeitspaketen, in denen Grid-Technologien für ein konkretes Szenario eingesetzt wurde.

Dieses Dokument ist wie folgt aufgeteilt. In Abschnitt 2 werden die Erfahrungen mit den Batch-Systemen Condor, Sun Grid Engine 6 und TORQUE/Maui dargestellt. Darauf folgen in Abschnitt 3 die Erfahrungen mit den Grid-Middlewares Globus Toolkit 4, UNICORE 5 und gLite. Die Erfahrungen mit den Grid-Technologien GridFTP und dCache zum Datenmanagement befinden sich in Abschnitt 4. Am Ende wird das Dokument mit einem kurzen Ausblick in Abschnitt 5 abgeschlossen.

2.Batch-Systeme

Bereits vor der Entstehung des Konzepts eines Computing-Grids wurde für die Verwaltung von lang dauernden, nicht-interaktiven und ggf. parallel ablaufenden Rechenaufträgen dedizierte Software entwickelt, die meistens in Rechenzentren und auf Supercomputern Einsatz fand. Diese Softwarekategorie, bekannt als Batch-Systeme, stellt auch heute einen wesentlichen Teil jeder Computing-Grid-Infrastruktur dar. Sie wird zur effizienten Lastverteilung, Ausführung und Überwachung von Anwendungen innerhalb der einzelnen zum Grid gehörenden Computing-Ressourcen verwendet, während die eigentliche Grid-Middleware sich um die Abwicklung entsprechender Aufgaben zwischen den Computing-Ressourcen konzentriert bzw. eine einheitliche Schnittstelle zu diesen nach außen anbietet.

Im Rahmen des Projekts WISENT wurde eine Auswahl von geeigneten Batch-Systemen für die bei den Projektpartnern vorliegenden Rechenressourcen unter Berücksichtigung ihrer besonderen Merkmale vorgenommen. Nach dieser Auswahl wurden die Batch-Systeme Condor und TORQUE produktiv eingeführt. Das Batch-System Sun Grid Engine 6 wurde zum Vergleich im Rahmen einer Diplomarbeit getestet.

Die Erfahrungen mit den Batch-Systemen Sun Grid Engine 6, Condor und TORQUE/Maui sind in den Abschnitten 2.1 bis 2.3 dargestellt. Auf Grund der umfassenderen Erfahrungen mit Condor und TORQUE/Maui im Produktivbetrieb auf dem Compute-Cluster im OFFIS, wird in Abschnitt 2.4 ein Vergleich zwischen diesen beiden Batch-Systemen vorgenommen. Abschließend wird in Abschnitt 2.5 umfangreich über die Erfahrungen mit Condor bei DLR PA berichtet.

2.1 Sun Grid Engine 6

Von der Sun Grid Engine (SGE) gibt es zum einen die kommerzielle Sun N1 Grid Engine 6 und zum anderen das Open-Source-Projekt unter der Sun Industry Standards Source Licence. Aufbauend auf der Open-Source-Version stehen auch aktualisierte Versionen der kommerziellen Version zur Verfügung. Die kommerzielle Version verfügt zusätzlich über einige Erweiterungen, die in der Open-Source-Version nicht verfügbar sind. Dazu gehören z. B. Programme zum Verwalten des Batch-Systems, die Möglichkeit, Computer mit Microsoft Windows in das Grid zu integrieren und ARCo, eine Software, um das Grid zu überwachen und zu analysieren. Inzwischen ist auch die kommerzielle Version kostenlos verfügbar.

Im Rahmen einer Diplomarbeit wurde die kommerzielle Variante der SGE in der Version 6.0 Update 8 eingesetzt und auf ihre Leistungsfähigkeit hin untersucht.

Die Sun Grid Engine wurde ursprünglich von Gridware in Regensburg in Deutschland entwickelt und hatte den Namen CODINE (Computing for Distributed Network Environments). Die Entwicklung von CODINE begann im Jahre 1992. Nach dem Aufkauf von Gridware im Jahre 2000 durch Sun Microsystems, wurde CODINE in Sun Grid Engine umbenannt und der Quelltext veröffentlicht.

Die Sun Grid Engine bietet von sich aus keine explizite Möglichkeiten für die Zusammenarbeit mit anderen Batch-Systemen oder Grid-Middleware. Über einen kommerziellen Adapter von GridwiseTech soll es möglich sein, dass die SGE mit dem Globus Toolkit und anderen Grid-Middleware-Lösungen zusammenarbeitet.

Die SGE benötigt ein Verzeichnis, z.B. ein über das Network File System (NFS) eingebundenes Verzeichnis, auf das alle Computer zugreifen können. Hier werden die Konfigurationen der einzelnen Hosts und des Batch-Systems gespeichert. Zusätzlich benötigt jeder Computer einen eindeutigen Hostnamen. Dieser wird während der Installation zur Computer-Identifikation genutzt. Die direkte Angabe einer IP wird vom Installationsprogramm abgelehnt und kann daher nicht verwendet werden.

Die SGE kann auf mehrere Arten installiert werden:

- Manuell
- Automatisiert
- Mit zusätzlichen Sicherheitsfunktionen

Bei der manuellen Installation wird zuerst das Installations-Skript für den Master Node (dem zentralen Computer, der die SGE-Installation verwaltet) auf dem entsprechenden Rechner gestartet. Nachdem der Master-Node installiert ist, wird das Installations-Skript für die so genannten Execution Hosts (Worker Nodes) auf den einzelnen Rechnerknoten gestartet. Sobald die ersten Execution Hosts eingerichtet sind, kann die SGE Jobs auf diese verteilen.

Je mehr Computer eingerichtet werden sollen, desto aufwändiger wird die manuelle Installation. Daher gibt es die Möglichkeit, die SGE automatisiert über eine Konfigurationsdatei zu installieren. Die automatisierte Installation installiert automatisch die verschiedenen Hosts auf den Computern und richtet bei Bedarf auch eine Datenbank zur Speicherung von Daten aus dem operativen Betrieb ein.

In der Hilfedatei zur automatisierten Installation steht, dass die Installation mit erweiterten Sicherheitsfunktionen nicht durch die automatisierte Installation unterstützt wird. In den Konfigurationsdateien können jedoch die Parameter für die Installation mit erweiterten Sicherheitsfunktionen gesetzt werden. Versuche, über die automatisierte Installation, die erweiterten Sicherheitsfunktionen zu installieren, schlugen allerdings fehl.

Die SGE kann automatisch wieder deinstalliert werden. Ein Deinstallieren über das Installations-Skript ist bei der manuellen und der automatisierten Installation möglich.

Gewöhnlich werden die Konfigurationen der Execution Hosts im globalen NFS-Verzeichnis gespeichert. Alternativ können sie auch jeweils lokal verwaltet werden. Das globale Verzeichnis muss aber in jedem Fall für die Konfiguration des Batch-Systems verfügbar sein.

Positiv bei der SGE ist zunächst die Installationsanleitung. In dieser wird Schritt für Schritt durch die Installationsmöglichkeiten der SGE geführt. Des weiteren ist die automatisierte Installation und Deinstallation sehr hilfreich, vor allem, wenn mehrere Konfigurationen automatisch nacheinander getestet werden sollen.

Für die Administration der SGE kann die Administratoranleitung viele vertiefende Informationen liefern. Was bei der Dokumentation allerdings teilweise fehlt, sind Erklärungen für Einstellungen und Parameter. Es wird erklärt, wie etwas eingestellt werden kann, jedoch fehlen Informationen über die Bedeutung der einzelnen Parameter. Trotz vieler gemeldeter Fehler wurde die offizielle Dokumentation seit 2005 nicht aktualisiert. Für spezielle Informationen und die Bedeutung der einzelnen Parameter ist daher das aktuelle Handbuch im CVS-Repository die beste Quelle.

Sehr positiv viel auf, dass durch die SGE automatisch die Prozessorkerne der Rechenknoten im Compute-Cluster im OFFIS erkannt und dann das Batch-System entsprechend eingerichtet wurde. So kann pro Prozessorkern ein Job ausgeführt werden und nicht nur ein Job pro Rechenknoten. Diese Einstellungen können für jeden Execution Host bei Bedarf manuell geändert werden.

Für die Diplomarbeit wurden mit der SGE zwei Batch-Systeme eingerichtet und mit jeweils zwei Anwendungen getestet. Durch die automatisierte Installation war ein schnelles und unkompliziertes Installieren der Sun Grid Engine möglich. Bei den Tests stellte sich heraus, dass die Sun Grid Engine für die untersuchten Anwendungen nahezu optimal die Last verteilt. Allerdings kann es bei einem ungünstigen Scheduling-Intervall in Bezug zur Jobdauer zu deutlichen

Verzögerungen bei der Gesamtlaufzeit kommen. Je länger die Jobs sind, desto weniger fallen diese Verzögerungen durch die Wahl des Scheduling-Profils ins Gewicht. Bei einem Vergleich der verschiedenen Konfigurationen stellte sich heraus, dass die Wahl des Scheduling-Profils den größten Einfluss auf die Gesamtlaufzeit hat. Hier war die Einstellung „max“ die beste Wahl. Dies liegt daran, dass das Scheduling-Intervall bei dieser Einstellung entsprechend kurz ist und nicht zu unnötigen Verzögerungen führt.

Im Rahmen der Diplomarbeit wurde gezeigt, dass die Sun Grid Engine ein Batch-System ist, das auch für den praktischen Einsatz in Desktop-Grids verwendet werden kann. Sie bietet eine ausführliche und gut strukturierte Dokumentation. Bei speziellen Problemen kann die aktive Mailingliste für Lösungshinweise genutzt werden. Durch die automatisierte Installation auf Basis von Konfigurationsdateien kann die Sun Grid Engine schnell und unkompliziert auf mehreren Computern installiert werden. Das Verwenden von Zertifikaten und verschlüsselten Verbindungen führt allerdings zu einem hohen Mehraufwand bei der Installation und Konfiguration, da diese Sicherheitsfunktionalität nicht mit der automatisierten Installation eingerichtet werden kann.

2.2 Condor

Condor ist ein Softwarepaket zur Realisierung des so genannten High-Throughput-Computing. Unter Nutzung vorhandener verteilter Rechenkapazitäten soll damit der Berechnungsdurchsatz (Anzahl von erfüllten, unabhängigen Aufträgen pro Zeiteinheit, z. B. pro Monat) in einer Organisation erhöht werden. Dieser Ansatz unterscheidet sich vom besser etablierten High-Performance-Computing. Im letzteren Fall wird die erforderliche Rechenkapazität innerhalb kürzester Zeit gefordert, und ihre effektive Nutzung erfordert Anpassungen in dem Anwendungscode. In vielen Anwendungskontexten, einschließlich dem Projekt WISENT, spielt jedoch HPC keine vorrangige Rolle.

Das Projekt Condor startete bereits 1988. Heute werden durch die Software alleine am Standort des Entwicklers (Computer Science Department, University of Wisconsin) mehrere hundert Rechner verwaltet. Weltweit existieren mehr als 1700 Installationen mit über 85000 Rechnern. Ursprünglich wurden für Condor lediglich Binaries angeboten, mittlerweile wurden aber auch der dazugehörige Quelltext veröffentlicht.

Der ursprüngliche Fokus des Projekts Condor lag auf der Lastverteilung in verteilten Systemen. Im Laufe der Zeit wurde das Thema etwas eingeschränkt und die Aufmerksamkeit dem „distributely owned computing“ gewidmet. Das Augenmerk ist die Durchführung von Berechnungsaufträgen mit Hilfe von vorhandenen Desktop-Arbeitsplätzen unter strenger Berücksichtigung von Nutzungsrichtlinien, die von jedem Anwender autonom festgelegt werden können. Die Idee ist, die Berechnungen nur dann ablaufen zu lassen, wenn sie die interaktive Nutzung der Maschine nicht beeinträchtigen (nachts, während Kaffeepausen usw.). Um dies zu gewährleisten, verfügt Condor über einen ausgefeilten Mechanismus zur Beschreibung der Verfügbarkeit von Rechenressourcen und zur Überwachung (und Unterbrechung) von Berechnungsaufträgen. Diese Flexibilität erlaubt Condor, heterogene Rechner zu

einem Verbund zusammenzufassen: Aspekte wie Rechnerarchitektur, Betriebssystem, Softwareausstattung, bevorzugte Nutzungsart usw. lassen sich administrativ beschreiben und für die Auftragsverteilung nutzen.

Im Projekt WISENT wurde die Condor-Einführung vor allem durch die Anforderungen bezüglich der einheitlichen Nutzung von verfügbaren Arbeitsplatzrechnern und des Computing-Cluster beim Projektpartner DLR PA motiviert. Vor dem Projektanfang wurden darauf lang dauernde Strahlungstransportrechnungen mit Hilfe eines ausgefeilten Lastverteilungsverfahrens (PVM/ppmake) auf die zum Rechencluster gehörenden Rechenknoten verteilt oder auf die zur Verfügung stehenden Arbeitsplatzrechner manuell kopiert und dort mit Hilfe von Shell-Skripten ausgeführt. Diese Vorgehensweisen hatten Nachteile technischer und organisatorischer Art. Beispielsweise war die Zuverlässigkeit von „ppmake“ eingeschränkt, und beim Fehlschlagen eines einzelnen Teilauftrags musste evtl. die gesamte Berechnung wiederholt werden, wodurch ggf. Rechen- und Arbeitszeit verschwendet wurden. Bei der Nutzung von Desktop-Arbeitsplatzrechnern für Berechnungen bestand die Gefahr, dass diese Aktivitäten die normale interaktive Verwendung der Rechner behindern. Außerdem konnten die einmal gestarteten Rechenaufträge nicht während ihrer Ausführung von einem Rechner auf einen anderen übergehen oder ihre Zwischenergebnisse sichern, wodurch wieder die Gefahr des Datenverlustes bei Abbruch gegeben war. Alle diese Probleme werden durch die eingebauten Condor-Funktionen behandelt, da Condor speziell für die Einrichtung von Desktop-Grids entworfen wurde.

Die Einführung von Condor bei DLR PA verlief erfolgreich mit Hilfe einer Installationsanleitung, die von OFFIS an Hand der offiziellen, umfangreichen Dokumentation erstellt wurde. Der Einsatz von Condor ermöglichte eine bequemere Ausführung von Strahlungstransportrechnungen einschließlich der Rechenzeit-aufwändigen 3D-Simulationen und ersetzte die vorherigen, suboptimalen Verfahren, besonders bei der Verwendung des internen Clusters. Insbesondere wurde auch das Checkpointing-Feature von Condor ausgenutzt, um die Ausführung von lang dauernden Jobs ohne Datenverlustgefahr zu realisieren. Die Erfahrungen von DLR PA, die noch genauer Abschnitt 2.5 beschrieben sind, führten dazu, dass die Software ebenfalls beim Projektpartner Universität Oldenburg für die Nutzung von Desktop-Rechnern installiert wurde.

Als problematisch stellt sich bei Condor vor allem die sehr umfangreiche Menge von Konfigurationseinstellungen dar, die die Funktionsweise der Software beeinflussen. Die meisten von ihnen werden zwar bereits bei der Installation mit sinnvollen Werten belegt und sind gut dokumentiert. Allerdings müssen zur Optimierung des Ressourcenmanagements und zur Problembehebung die Zusammenwirkung von Konfigurationseinstellungen und die von Condor verwendeten Algorithmen verstanden werden. Da diese durch Anwender nicht einfach intuitiv nachvollziehbar sind, wurden die Abweichungen zwischen dem tatsächlichen und erwarteten Verhalten von ihnen als Fehler empfunden, obwohl sich Condor gemäß der konfigurierten Vorgaben verhielt. Beispielsweise wurden die laufenden Rechenaufträge unerwartet bzw. zu oft suspendiert; dieses Problem ergab sich aus Konfigurationseinstellungen und konnte durch einen Eingriff des Administrators behoben werden. Ein weiteres Problem ergab sich durch die Erschöpfung des freien Festplattenspeichers auf der Maschine, von der die Jobs

abgeschickt wurden. Dieses Problem äußerte sich darin, dass die Rechenknoten als belegt erschienen, obwohl die Jobs ausdrücklich gestoppt wurden. In dieser Situation zeigte Condor leider unzureichende Fehlertoleranz und mangelhafte Fehlerberichterstattung. Die Problembehebung war dagegen nach der Feststellung seiner Ursache einfach.

Eine Anforderung, die erst nach der Condor-Einführung und einiger Verwendungszeit deutlich wurde und nur zum Teil zufriedenstellend erfüllt werden konnte, ist die relative Priorisierung von Benutzern bzw. Benutzergruppen. Condor versucht zwar, eine faire Zuweisung von Rechenressourcen nach den vorgegebenen Prioritäten zu erreichen. So kann man z. B. administrativ festlegen, dass ein Benutzer doppelt so viele Rechner wie ein anderer Benutzer bekommt. Leider werden solche Vorgaben durch Condor basierend auf der Gesamtanzahl aller Rechner umgesetzt, anstatt auf der aktuell verfügbaren, die Job-Anforderungen erfüllenden Anzahl. So kann sich im Fall der Ressourcenknappheit ergeben, dass dem Benutzer mit der höheren Priorität alle verfügbaren Rechner zugeordnet werden und dem Benutzer mit der niedrigeren Priorität gar keine, weil der bevorzugte Benutzer noch nicht die ihm zustehende Gesamtanzahl von Rechnern erschöpft hat.

Insgesamt ist Condor auf die Optimierung des Gesamtdurchsatzes in einer Umgebung mit autonom verwalteten Rechnern ausgelegt und für eine feingranulare, zentralisierte Festlegung von globalen Nutzungsrichtlinien eher ungeeignet. Dieser Mangel ging aus der Condor-Dokumentation nicht hervor, sondern konnte erst durch Experimente und Diskussionen in der Mailing-Liste verstanden werden. Es handelt sich hierbei leider nicht um ein fehlendes Feature, das möglicherweise in der Zukunft nachimplementiert wird, sondern vielmehr um grundsätzliche Beschränkungen, die sich aus dem verteilten, dezentralen Charakter des Scheduling-Algorithmus und aus fehlenden Informationen über den genauen Zustand aller Ressourcen ergeben.

2.3 TORQUE/Maui

Mit der Anschaffung eines Computing-Clusters im Rahmen der D-Grid-Sonderinvestition bei OFFIS musste ebenfalls ein geeignetes Batch-System installiert werden. Die vorherigen Erfahrungen mit Condor haben gezeigt, dass dieses zum Zeitpunkt der Cluster-Anschaffung bereits bei DLR PA und der Universität Oldenburg im Einsatz befindliche Batch-System zwar auch für OFFIS ausreichend wäre, aber aufgrund seiner Merkmale nicht optimal für die Verwaltung eines zentral administrierten D-Grid-Clusters mit einer fixen Anzahl von Rechenknoten erscheint. Die wichtigsten Argumente für die Auswahl einer Alternative zu Condor waren:

- Relativ geringe Verbreitung und kleine Erfahrungsbasis für den Einsatz von Condor im HPC-Bereich
- Fehlende Unterstützung für Condor seitens des D-Grid-Integrationsprojekts (DGI) und Wunsch nach einer Vereinheitlichung der Batch-Systeme im D-Grid
- Fehlende Condor-Features, die ein besseres zentrales Cluster-Management ermöglichen würden (z. B. Vorab-Reservierung von Rechenzeit für bestimmte Benutzergruppen)

- Bessere Integrationsaussichten mit zukünftigen D-Grid-Accounting-Systemen für TORQUE/Maui.

- Überflüssige Condor-Features, wie z. B. Erkennung der interaktiven Rechnernutzung oder die Migration von laufenden Jobs zwischen Rechnern. Diese Features werden im Kontext eines Rechenclusters (im Gegensatz zum Desktop-Grid) nicht benötigt und tragen zur Komplexität bei.

TORQUE ist ein Batch-System, das als Nachfolger von PBS eine lange Entwicklungsgeschichte im High-Performance-Computing und infolgedessen auch eine entsprechend hohe Stabilität aufweist. In Kombination mit dem Scheduler Maui bietet es eine weit verbreitete und offene Lösung in seinem Einsatzbereich. Als Option wird von den Entwicklern auch der Migrationsweg zum kommerziellen Cluster-Management-Produkt Moab angeboten.

Die Installation von TORQUE/Maui verlief bei OFFIS nach der Anleitung, die vom DGI geliefert wurde. Die offizielle Dokumentation für die Produkte ist im Vergleich zu Condor weniger umfassend und schlechter gepflegt. Zum Teil wird auf die nicht mehr aktuelle Dokumentation von PBS bzw. OpenPBS verwiesen.

Bei der Installation von Maui traten Probleme auf, die durch eine mangelhafte Verpackung der Binary-Version in ein RPM-Paket für das Betriebssystem SuSE Linux Enterprise Server 10.0 verursacht wurden. Für diese Probleme wurde in WISENT eine Lösung erarbeitet und an DGI berichtet. Dort wurden entsprechende Korrekturmaßnahmen ergriffen, von denen auch D-Grid-Projekte profitieren konnten.

Während des initialen Cluster-Betriebs ist in TORQUE ein weiteres Problem aufgetreten, das durch den Festplattenplatzmangel auf der Root-Partition von Rechenknoten verursacht wurde und sich dadurch manifestierte, dass von den ausgeführten Jobs keine Ausgaben auf den zentralen Rechner übertragen wurden. Die sichtbaren Symptome dieses Problems, wie es auch bei dem ähnlichen Problem mit Condor bei DLR PA der Fall war, wiesen nicht auf seine Ursache hin. Die Problembehebung war nach der Feststellung der Ursache jedoch leicht.

Da der OFFIS-Cluster verschiedenartige Rechenknoten enthält (mit 4 oder 2 Prozessoren, mit oder ohne InfiniBand-Netzwerkkarte), wurden die Rechenknoten mit Hilfe von zusätzlichen Attributen kategorisiert. Diese Attribute können in einer Job-Beschreibung verwendet werden, um den Job entsprechend zu einem Knoten mit der gewünschten Ausstattung zu lenken. Allerdings fiel dabei auf, dass die Job-Beschreibungssprache von TORQUE im Vergleich zu Condor viel beschränkter ist, da ausschließlich statische, administrativ definierte Attribute und ausschließlich einfache Vergleiche unterstützt werden.

Neben der Ausführung von sequentiellen Jobs unterstützt TORQUE in Kombination mit dem Programm „mpiexec“ die Ausführung von parallelen Jobs, die mit Hilfe der Bibliothek MPI miteinander kommunizieren. Im OFFIS-Cluster wurde die MPICH-Implementierung aus dem Lieferumfang des kommerziellen PGI-Compilers installiert. Zusätzlich wurde für die Nutzung der InfiniBand-Netzwerkkarten, die eine deutlich geringere Latenzzeit und höhere Bandbreite als das von MPICH verwendete Gigabit Ethernet aufweisen, die MVAPICH-Implementierung aus dem Treiberpaket OFED (OpenFabrics Enterprise Distribution) der Firma Mellanox installiert.

Die Nutzung von MVAPICH mit TORQUE stellte sich zum Anfang als unmöglich heraus. Der Grund dafür war ein Fehler im Hilfsprogramm „mpiexec“. Dieser konnte in Zusammenarbeit mit seinem Entwickler behoben werden. Mit dieser Fehlerkorrektur leistete WISENT einen Beitrag für die internationale HPC-Community, die ebenfalls an der Nutzung von TORQUE in Kombination mit der aktuellen Version der InfiniBand-Treiber interessiert ist.

2.4 Vergleich zwischen Condor und TORQUE/Maui

Aufgrund von gesammelten Erfahrungen mit Condor und TORQUE wurde ein Vergleich der beiden Batch-Systeme möglich, der einen Beitrag zum besseren Verständnis und einer fundierten Entscheidung zwischen diesen Systemen leistet. Die allgemein verfügbare Dokumentation beschränkt sich stets auf Merkmale des einen oder anderen Batch-Systems, ohne dass ein solcher Vergleich aus der Sicht der Softwaretechnik angeboten wird. Um diese Lücke zu schließen, werden an dieser Stelle die Unterschiede zwischen Condor und TORQUE knapp zusammengefasst. Sie lassen sich gut in zwei Kategorien unterteilen: Entwurfsunterschiede und Annahmen über den Verwendungskontext.

Die Entwurfsunterschiede bestehen in der Art der Trennnung des Ressourcenmanagement (Feststellung der Rechnerzustände, sowie die Job-Steuerung und -Ausführung) und Scheduling (Festlegung der Ausführungsreihenfolge) in Condor und TORQUE.

Condor besitzt eine monolithische Architektur, in der die Aspekte des Ressourcenmanagement und Scheduling zusammenwachsen. Im Umfang von Condor werden sowohl der Ressourcenmanager (zuständig für die Job-Steuerung und -Ausführung) als auch der Scheduler (zuständig für Entscheidungen über die Ausführungsreihenfolge) geliefert, und es gibt keine Möglichkeit für den Administrator, eine dieser Softwarekomponenten zu ersetzen.

Die Trennung zwischen dem Ressourcenmanagement und Scheduling ist dagegen sehr deutlich im Fall von TORQUE/Maui. TORQUE übernimmt in erster Linie die Aufgaben des Ressourcenmanagers. Das Softwarepaket enthält zwar einen rudimentären First-In-First-Out-Scheduler und ist damit auch alleinstehend verwendbar. Allerdings wird auch eine Scheduler-Schnittstelle angeboten, die eine Ersetzung des einfachen Schedulers durch eine ausgefeiltere Variante – Maui oder Moab erlaubt.

Die unterschiedlichen Annahmen über den Verwendungskontext von Condor und TORQUE/Maui lassen sich folgendermaßen charakterisieren:

Condor nimmt ein verteiltes Eigentum und Management von Ressourcen an (typisch für Desktop-Grids). Es werden folgende Rollen unterschieden: Condor-Administratoren, Ressourceneigentümer und Ressourcennutzer, wobei die letzten beiden Rollen oft von einzelnen Personen gemeinsam wahrgenommen werden. Unter diesen Annahmen werden Entwurfsziele definiert, die variierende Verfügbarkeit von Ressourcen zu tolerieren und die maximale Flexibilität bezüglich der Bestimmung der Ressourcennutzungsart für die Ressourceneigentümer zu gewährleisten. Charakteristisch ist, dass die Ressourceneigentümer jedoch keine Verpflichtungen für die Sicherstellung von Qualitätseigenschaften gegenüber Nutzern eingehen und nicht als Dienstleister auftreten. Die jeweils freien

Rechenkapazitäten werden lediglich vermittelt und den Nutzern ohne Garantie angeboten.

TORQUE/Maui nimmt ein zentralisiertes Eigentum und Management von Ressourcen an (typisch für HPC-Cluster in Rechenzentren). Es werden folgende Rollen unterschieden: Cluster-Administratoren, Cluster-Nutzer und Geschäftsführung. TORQUE/Maui hat zum Ziel, die maximale Flexibilität für einen Cluster-Administrator in der Festlegung von Nutzungsrichtlinien für wohl definierte Gruppen von Nutzern zu erreichen. Gleichzeitig sind die Administratoren eines Clusters dafür verantwortlich, vorhersehbare Qualitätseigenschaften für die Nutzergruppe zu gewährleisten und die allgemeinen Richtlinien und Prioritäten entsprechend den vorgegebenen Geschäftszielen durchzusetzen.

2.5 Einsatz von Condor bei DLR PA

2.5.1 Motivation

In der Abteilung für Fernerkundung der Atmosphäre bei DLR-PA werden unterschiedliche Aspekte der Wolken-Strahlung-Wechselwirkungen untersucht. Einer davon, der eine wichtige Rolle in vIEM spielt, ist die Entwicklung, Bewertung und Validierung von Verfahren zur Bestimmung von Solarstrahlung am Erdboden aus Satellitenbeobachtungen. Wolken weisen nämlich die höchste zeitliche und räumliche Variabilität auf und haben den größten Einfluss auf die Solarstrahlung am Boden. Dazu werden detaillierte Strahlungstransfersimulationen mithilfe des libRadtran (library for radiative transfer [Mayer und Kylling 2005]) Softwarepakets durchgeführt. Wolken- und Atmosphärendaten des LM (Lokalmodell) Vorhersagemodells des Deutschen Wetterdienstes (DWD) und Zusatzdaten zur Beschreibung des Bodenreflexionsvermögens werden als Input dafür benutzt, um Satellitenmessungen von MSG/SEVIRI und einfallende Solarstrahlung am Boden zu simulieren. Nach Anwendung der Solarstrahlungsalgorithmen auf diese berechneten Satellitenmessungen können die Resultate mit den ebenfalls berechneten Solarstrahlungsdaten verglichen werden, um die Genauigkeit der Retrievalverfahren zu bestimmen. Außerdem hat man dann die Möglichkeit, die Fälle im Detail zu untersuchen, welche erhebliche Abweichungen zeigen, weil alle Inputfaktoren, d.h. alle atmosphärischen Parameter, bekannt sind. Damit können diese Algorithmen verbessert werden.

Zur Berechnung der Satellitenbeobachtungen sowie der einfallenden Solarstrahlung am Boden für eine gewisse Region gibt es grundsätzlich zwei Möglichkeiten. Üblicherweise wird die Gesamtsimulation in einzelne Atmosphärensäulen zerlegt und angenommen, dass diese horizontal homogen und unabhängig voneinander sind. Unter diesen Annahmen können sogenannte eindimensionale (1D) Strahlungstransferrechnungen durchgeführt werden. Auf libRadtran aufbauend ist zu diesem Zweck bei DLR-PA das SIMSAT (Satellite Simulator) Modul geschrieben worden. Wenn hingegen keine Vereinfachungen gemacht werden und Photonen auf ihrem Weg durch die Atmosphäre genau verfolgt werden, sind diese sogenannten dreidimensionalen (3D) Strahlungstransferrechnungen aufwändiger und werden mit dem MYSTIC (Monte

Carlo code for the physically correct tracing of photons in cloudy atmospheres [Mayer 2000]) Modul, auch auf libRadtran basierend, durchgeführt.

Wenn Satellitenbeobachtungen für ausgedehnte Gebiete simuliert werden, sind schon 1D Rechnungen sehr rechenzeitintensiv. Bisher wurden solche Arbeiten entweder auf einzelnen Standard-Arbeitsplatzrechnern unter Linux/UNIX oder auf einem 32 x 2 GHz PC-Cluster durchgeführt. Für einzelne Anwendungen wurden auch die gerade außerhalb der täglichen Kernarbeitszeiten wenig genutzten Arbeitsplatzrechner für verteiltes Rechnen genutzt. Allerdings wurde dies bisher durch personalintensives einzelnes Ansprechen der Rechner vorgenommen, eine automatische technische Lösung bestand nicht. Deswegen wurde im Rahmen von WISENT die Vernetzung der einzelnen Knoten des Linux-Clusters sowie die Vernetzung von 30 Arbeitsplatz-PCs mit Hilfe der Condor-Software vorgenommen.

2.5.2 Condor Installation und MYSTIC

Die Installation des Condor-Paketes mit der vorbereiteten Anleitung von OFFIS bereitete keine Probleme. Testläufe mit einfachen kleinen Beispielprogrammen, die im Paket enthalten sind, verliefen ebenfalls ohne Störungen.

Zum Testen der Berechnung von rechenzeitintensiveren Anwendungen wurden 25 Jobs mit dreidimensionalen Strahlungstransferrechnungen vorbereitet, wobei diese sich in einigen atmosphärischen Parametern unterschieden, um die Berechnung eines realistischen Wolkenfeldes zu simulieren. Alle Jobs wurden auf einer Maschine mit 3 GHz CPU bereits im Vorfeld berechnet. Die Dauer der Berechnungen variierte zwischen 4 Stunden und 85 Stunden.

Die Jobs wurden zunächst unter Verwendung des „vanilla“-Universums an das Condor-Netzwerk übergeben. Die Verteilung der Jobs auf die vorhandenen Knoten lief problemlos. Folgende Beobachtungen konnten während der Tests gemacht werden:

- Bei Simulation von mehreren Knoten unter Verwendung von hyperthreading auf Servern wird der Gesamtspeicher auf die entstehenden virtual machines aufgeteilt. Jede virtual machine erscheint als unabhängiger Knoten mit einer begrenzten Kapazität an Speicher im Condor-Netzwerk. Dies kann dazu führen, dass bei Angabe von einem Mindestspeicher in den Condor-requirements der betreffende Rechner nicht mehr verwendet wird, obwohl er eigentlich verfügbar wäre und auch entsprechende Voraussetzungen bzgl. CPU und Speicher hätte. Dieses Problem wurde gelöst, indem nur eine begrenzte Anzahl an virtual machines pro Rechner in das Condor-Netzwerk eingetragen wurden.

- Die Definition von „idle“ innerhalb Condors erwies sich während der Testläufe als zu weit gefasst, da Mausbewegungen beispielsweise beim Öffnen eines Browsers die Condor-Jobs nicht stoppen. Erst durch Tastatureingaben oder ssh-Zugriff werden Jobs angehalten. Dies konnte dazu führen, dass ein Rechner auf die Eingaben des Users sehr schwerfällig reagierte.

- Startet ein User sehr viele Jobs, ist das gesamte Netzwerk solange für andere User blockiert, bis diese Jobs abgearbeitet sind. Es können zwar über „user priorities“ einzelne User vorrangig behandelt werden, eigentlich sollte dies aber nicht der Fall sein. Des Weiteren können über die Angabe von Rank Einstellungen vorgenommen werden, wie genau Jobs zu verteilen sind. Nachteil ist aber, dass dies vom User im Condor command file angegeben werden muss. Der User muss also einerseits wissen, wie er die Einstellungen am besten konfiguriert und andererseits auch bereit sein, seinen eigenen Job nicht mit maximaler Ausdehnung im Netz laufen zu lassen. Wünschenswert wären Einstellungen, die vom Administrator bei der Einrichtung des Netzwerkes vorgenommen werden können, z.B. maximale Anzahl von Jobs pro User.

- Jobs auf Rechnern, die als Arbeitsplatzrechner dienen, wurden oft in kurzen Abständen suspended und wieder unsuspended. Zum Teil wurden die Jobs schließlich nach mehreren Stunden abgebrochen und neu verteilt, wobei sie jedoch häufig wieder auf den Rechner geschoben wurden, auf dem sie gerade abgebrochen worden waren. Die Berechnung dieser Jobs dauerte dementsprechend sehr lange. Um die Verteilung der Jobs zu verbessern und auch dem suspend/unsuspend vorzubeugen, wurde die ContinueIdleTime auf 5 Minuten hochgesetzt und ein rank=kflops in die command-Datei eingefügt, um bei der Neuverteilung schnelle Rechner zu bevorzugen.

Um bei Benutzung des DLR-PA-Netzwerkes zu gewährleisten, dass nach stundenlanger Bearbeitung eines Jobs nicht alle Ergebnisse verloren gehen, wenn ein User an seinen Arbeitsplatzrechner zurückkehrt, wurde die Verwendung des „standard“-Universums und damit die Möglichkeit des Checkpointing in Betracht gezogen. Kleinere C-Programme konnten unter Verwendung von condor_compile kompiliert und verlinkt werden. Für das Gesamtpaket von libRadtran erwies sich dies jedoch als äußerst schwierig. Um überhaupt einen fehlerfreien Kompilierungsvorgang mit condor_compile durchführen zu können, mussten einige Anpassungen in der betreffenden libRadtran-library vorgenommen werden. Als Hürde erwies sich auch die Einbeziehung der netcdf library. Der Pfad für diese kann zwar beim Ausführen des entsprechenden configure-Skriptes explizit angegeben werden, so dass sich libRadtran unter normalen Umständen problemlos kompilieren lässt. Unter Einbeziehung von Condor führte dies jedoch nicht zum Erfolg. Eine vollständige Kompilierung konnte nur erreicht werden, wenn die netcdf-library im System statisch verlinkt war, aber auch dann nicht auf jedem System. So enthalten SuSE-Versionen ab 10.x zwar netcdf, führen aber trotzdem zu Fehlermeldungen und Abbruch der Kompilierung. Funktionierte die Ausführung des make-Befehls, war die Dauer des Kompilierungsvorganges sehr stark erhöht. Gegenüber Kompilation ohne Einbeziehung von Condor verlängerte sich diese mindestens um einen Faktor 5.

Noch ein Faktor musste berücksichtigt werden: wenn libRadtran mit Condor kompiliert wird und im „standard“-Universum arbeitet, hat MYSTIC keinen Zugriff auf den Rechner, wo der Job tatsächlich läuft, weil die Umgebung des Rechners

simuliert wird, wo der Job eingereicht wurde. Das macht die Abfrage von host id und ähnlichem unmöglich, was aber einen Einfluss auf die Erzeugung der Zufallszahl hat, die zum Initialisieren des Monte Carlo Prozesses gebraucht wird. Fangen zwei MYSTIC-Rechnungen mit derselben Zufallszahl an, sind die Resultate identisch. Das Problem kann man umgehen, indem man diese Zufallszahl an MYSTIC explizit übergibt.

Auf dem DLR-PA-Cluster war die Kompilation von libRadtran nach den oben genannten Veränderungen problemlos möglich. Auch die Bearbeitung von Jobs unter Verwendung des „standard“-Universums funktionierte auf diesem System. Im Abstand von 3 Stunden wurden Checkpoints angelegt und bei Bedarf der Job auf einen anderen Knoten verlagert.

2.5.3 SIMSAT

Als zusätzliche Anwendung wurde das SIMSAT Paket zur Simulation von Satellitenstrahldichten mit eindimensionalen Strahlungstransferlösern mit den Condor-Funktionalitäten erweitert. Der technische Hauptunterschied zu den oben erwähnten dreidimensionalen Rechnungen besteht darin, dass SIMSAT kein C-Programm ist, sondern ein C-Shell-Skript, das unter Linux läuft und unterschiedliche C-Programme und weitere Shell-Skripte aufruft. Unter den C-Programmen befindet sich natürlich auch libRadtran zur Lösung der Strahlungstransportgleichung. Besondere Aufmerksamkeit musste hier den temporären Dateien gewidmet werden, die von SIMSAT während eines Laufes erzeugt werden und Gefahr laufen, von konkurrierenden SIMSAT-Läufen überschrieben zu werden. Mit einer sauberen Namensgebung und den mit MYSTIC gesammelten Erfahrungen konnte SIMSAT mit Condor im „vanilla“-Universum auf den DLR-PA-Arbeitsplatzrechnern und auf dem DLR-PA-Cluster benutzt werden. Unterschiedliche Tests wurden dabei erfolgreich durchgeführt.

Schließlich wurde die Nutzung des von OFFIS betreuten Compute-Clusters mit Condor-GT4 angestrebt und realisiert. Nach erfolgreicher Beantragung eines Grid-Zertifikats erwies sich die DLR-Firewall als das Hauptproblem. Mit Hilfe von OFFIS konnten diese Schwierigkeiten beseitigt werden, indem

- Ein Rechner außerhalb der eigentlich Firewall in einer so genannten demilitarisierten Zone (DMZ) als „submission host“ eingesetzt wurde.
- GridFTP die erlaubten Ports für den benötigten Datentransfer explizit mitgeteilt wurden.
- Die genaue Identifikation des DMZ-Rechners seitens des OFFIS-Clusters sichergestellt wurde.

Die für das SIMSAT-Paket notwendigen Veränderungen lassen sich wie folgt zusammenfassen:

- Der „submission host“ und die CPUs, die die Strahlungstransferrechnungen durchführen, verfügen nicht über einen gemeinsamen Speicher, auf den sie zugreifen könnten. Deswegen musste auf der einen Seite gewährleistet werden, dass alle nötigen Files dem OFFIS-Cluster zur Verfügung stehen; und auf der anderen Seite, dass die unterschiedlichen Verzeichnisstrukturen auf dem DMZ- und den OFFIS-Rechnern berücksichtigt wurden.

- SIMSAT wurde so angepasst, dass Grid-Ressourcen wie das oben erwähnte GridFTP-Client zum Einsatz kommen, wenn Input- und Output-Daten von und zu dem einreichenden DMZ-Rechner transferiert werden.

- Wegen der kurzen Rechenzeit eines einzelnen Jobs und der gleichzeitig hohen Anzahl solcher Jobs, erwies es sich als sehr nützlich, das von OFFIS entwickelte Perl-Modul Multijob zu benutzen, das mehrere Jobs bündelt und sie auf eine beschränkte Anzahl Prozessoren verteilt. Damit konnte erreicht werden, dass die NFS-Last auf dem OFFIS-Cluster begrenzt blieb.

Das so modifizierte SIMSAT-Paket wurde zusammen mit libRadtran letztendlich „per Hand“ auf dem OFFIS-Cluster installiert und dort mittels Condor-GT4 vom „submission host“ angesprochen.

3.Grid-Middleware

Neben Batch-Systemen, die ein wichtiger Bestandteil einer Grid-Infrastruktur sind, basiert die eigentliche Konstruktion eines Grid im Wesentlichen auf dem Einsatz von Grid-Middleware. Eine Grid-Middleware abstrahiert von den Ressourcen und den unterschiedlichen Zugriffsmöglichkeiten, z.B. bei Compute-Ressourcen von den unterschiedlichen Batch-Systemen, und bietet eine einheitliche Zugriffsschnittstelle an. Zudem finden Sicherheitsaspekte, z.B. Zugriffssicherheit durch eine „Authentication and Authorisation Infrastructure“ (AAI), eine besondere Berücksichtigung.

Im Rahmen des Projektes WISENT wurden zunächst verschiedene Grid-Middleware gesichtet und grob verglichen. Die Entscheidung fiel aus unterschiedlichen Gründen recht schnell auf Globus Tookit 4. Hauptgründe waren der hohe Bekanntheitsgrad, die vielfältigen Funktionen für das Datenmanagement, und die Verwendung von Grid-Standards wie das „Web Services Resource Framework“ (WSRF). Später wurde auf dem Compute-Cluster im OFFIS neben Globus Toolkit 4 noch UNICORE 5 und gLite installiert, um eine Vorgabe des DGI zum Betrieb der Sonderinvestitionen umzusetzen.

Die Erfahrungen mit Globus Toolkit 4, UNICORE 5 und gLite werden in den Abschnitten 3.1 bis 3.3 vorgestellt.

3.1 Globus Toolkit 4

Im Jahre 1995 wurden in den USA im Rahmen des Projektes I-Way 17 Hochleistungsrechenzentren miteinander vernetzt. Dies gilt als Startpunkt für die

Entwicklung des Globus Toolkit zur Vernetzung von verteilten Ressourcen, dessen Entwicklung von Ian Foster und Carl Kesselman, die den Begriff „Grid“ im Jahre 1998 prägten, vorangetrieben wurde. Seit dem Jahr 2002, das Globus Toolkit 2 hatte schon einen hohen Verbreitungsgrad erreicht, wird die Einführung von Standards im Grid-Umfeld fokussiert, um eine Interoperabilität mit anderer Grid-Middleware zu erreichen bzw. auf bereits bestehenden und etablierten Standards aufzusetzen. Diese Bestrebung wurde in der so genannten „Open Grid Services Architecture“ (OGSA) festgehalten, die insbesondere auf Web Service-Technologien aufbaut. Eine der daraus resultierenden Standards ist das „Web Services Resource Framework“ (WSRF), das in Globus Toolkit 4 (GT4) implementiert wurde. GT4 wurde als Grid-Middleware im Projekt WISENT ausgewählt. Dazu waren folgende Gründe ausschlaggebend.

- Hoher Bekanntheitsgrad
- Viele Dienste zum Datenmanagement z.B. GridFTP
- Verwendung von Standards wie WSRF
- Verfügbarkeit als Open-Source-Software
- Erweiterbarkeit
- Anbindung von Condor als Batch-System möglich

GT4 sollte im Projekt WISENT in erster Linie zur Vereinheitlichung von Datentransfers genutzt werden sowie als Schnittstelle zur Nutzung verteilter Compute-Ressourcen. Initial wurde das GT4.0.2 verwendet, worauf sich der folgende Text auch im Wesentlichen bezieht. Später wurde dann auf neuere Versionen gewechselt, insbesondere auf dem Compute-Cluster im OFFIS.

Für die erste Installation wurde das Red Book „Introduction to Grid Computing“ von IBM genutzt, weil darin sehr gut Schritt für Schritt die Installation des GT4 mit drei Rechnern (einer dient als CA) sowie einfache Tests beschrieben werden. Später wurde die offizielle Dokumentation auf www.globus.org konsultiert. Allerdings sind die dort angebotenen Dokumentationen der einzelnen Dienste zum Teil unvollständig bzw. recht knapp gehalten. Auf Basis dieser Quellen wurde eine eigene Installationsanleitung für die WISENT-Projektpartner erstellt. Bei jedem Projektpartner wurde das GT4 installiert. Zur Konfiguration wurden später die Vorgaben des DGI umgesetzt.

Die Installation kann entweder mit der Binary-Version oder der Source-Version durchgeführt werden. Binary-Installer gibt es für zahlreiche Linux-Betriebssysteme wie Red Hat, den Red Hat-Ableger Fedora, SuSE und Debian. Für alle anderen Unix/Linux-Systeme steht dann der Source-Installer zur Verfügung. Der Source-Installer hat allerdings den Nachteil, dass eine vollständige Installation gut eine Stunde dauern kann, je nach Performanz des Zielsystems. Der Binary-Installer hingegeben benötigt ein paar Minuten. Der Java-WS-Core, im Prinzip der WSRF-Grid Service-Container und mitgelieferte Grid Services, ist auch unter Windows lauffähig, weil dieser komplett in Java implementiert wurde. Weitere Komponenten wie der GridFTP-Server sind in C implementiert und laufen nur auf Unix/Linux-Systemen.

Die Installation des GT4 (4.0.2) hat einige Systemvoraussetzungen, wie Java SDK ab Version 1.4.2, Apache ANT ab Version 1.5.1, PostgreSQL ab Version 7.1, gcc ab Version 2.95, zlib ab 1.1.4, GNU tar/sed/make und sudo. Diese wurde mit Folgeversionen des GT4 jeweils angepasst. Unter Sun Solaris 9, was beim DLR eingesetzt wird, gab es dafür zum Teil keine vorgefertigten Pakete, so dass eine Installation bzw. Kompilierung vom Quelltext notwendig war, zum Teil mit weiteren Abhängigkeiten. Dies erzeugte einen größeren Mehraufwand zur Erfüllung der Systemvoraussetzungen, bevor die eigentliche Installation des GT4 durchgeführt werden konnte. Der Installationsvorgang selbst lief dann ohne Probleme ab. Mittlerweile ist aber auch eine Binary-Version für Solaris 9 und 10 verfügbar.

Die Einrichtung der PostgreSQL-Datenbank für den „Reliable File Transfer"-Dienst hat zum Teil Probleme verursacht. Zum einen lag das daran, dass sich zwischen den PostgreSQL-Versionen 7 und 8 die Konfigurationseinstellung geändert hat, mit der die Netzwerkschnittstelle der Datenbank aktiviert werden kann. Zum anderen sind Probleme mit Datenbankberechtigungen aufgetreten, weil die Datenbank mit einer falschen Benutzerkennung angelegt wurde, was auf eine schlechte Vorgehensweise zur Einrichtung einer Datenbank in der IBM-Anleitung zurückzuführen war.

Allgemein fiel bei der Einrichtung verschiedener Dienste negativ auf, dass für jeden Dienst, der eine Datenbank benötigt, eine eigene Datenbankverbindung konfiguriert wird. Globale „Connection-Pools" sind nicht vorgesehen. So nutzt der RFT-Dienst (Reliable File Transfer) beispielsweise eine Datenbank (z.B. PostgreSQL oder MySQL) über JDBC und der RLS-Dienst (Replica Location Service) über ODBC.

Bezüglich des Datenmanagements wurden mehrere Dienste getestet. Dazu gehören insbesondere GridFTP und der Reliabe File Transfer (RFT).

Bezüglich GridFTP stehen im WISENT-Kontext drei Optionen im Vordergrund:

- Unterstützung der Grid Security Infrastructure (GSI), so dass eine einmalige (single-sign on - SSO), auf X.509-Zertifikaten basierende Anmeldung möglich ist.
- Unterstützung von so genannten „Third Party Transfers", d.h. ein Client-Rechner baut eine Kontrollverbindung zu zwei GridFTP-Servern auf, woraufhin zur Datenübertragung eine direkte Verbindung zwischen den beiden GridFTP-Server aufgebaut wird. Der Datenstrom geht somit nicht über den Client selbst.
- Nutzung von mehreren, parallelen Datenkanälen, um höhere Durchsatzraten zu ermöglichen.

Erste Tests mit GridFTP haben gezeigt, dass strenge Firewall-Regeln dessen Nutzung stark beeinträchtigen bzw. gänzlich verhindern können. Grund ist die Verwendung von dynamischen Ports zur Öffnung von Datenkanälen, wobei vom DGI die pauschale Öffnung des Portbereichs 20000-25000 empfohlen wurde. Das war insbesondere mit den Sicherheitsrichtlinien beim DLR nicht kompatibel. Weitere Informationen dazu sind im Dokument „AP 2.1 - Bericht zu den Auswirkungen der DLR-IT-Sicherheitsstrukturen beim Datentransfer mit Partnern außerhalb des DLR" zu finden.

Üblicherweise sind bei GridFTP der Kontrollkanal verschlüsselt und die Datenkanäle unverschlüsselt. Der Aufbau des Datenkanals wird aber über eine so genannte „data channel authentication“ (DCAU) abgesichert. Über entsprechende Optionen kann der Datenkanal zusätzlich gegenüber Manipulation (Checksumme) abgesichert oder gar komplett verschlüsselt (Checksumme und Verschlüsselung) werden, was aber bei Tests zu der erwarteten starken Reduzierung (um mehr als die Hälfte) des Datendurchsatzes geführt hat.

Im Rahmen kleinerer Datentransfertests mit GridFTP wurde für ein Teil der in WISENT verwendeten Datentransferwege überprüft, inwieweit die Nutzung paralleler Datenkanäle den Durchsatz erhöhen können. Die Ergebnisse dazu befinden sich in Abschnitt 4.1.1.

Der „Reliabe File Transfer“ (RFT) des GT4 kann dazu genutzt werden, mehrere Datentransfers durchzuführen und überwachen zu lassen. Der Zustand der einzelnen Datentransfers wird dabei in einer Datenbank persistiert. Mit dem RFT-Dienst können nur so genannte „Third-Party-Transfers“ mittels GridFTP durchgeführt werden. Bis auf die zusätzliche Absicherung der Datenkanäle sind alle GridFTP-Optionen nutzbar. Allerdings verwendet der RFT-Dienst eine umständlich aufgebaute Eingabe-Datei mit fester Struktur, in der am Anfang Werte für bestimmte Optionen in einer bestimmten Reihenfolge untereinander stehen müssen, gefolgt von Quell-Ziel-Paaren für die zu übertragenden Dateien. Ein einzelner RFT unterstützt dabei nur Transfers mehrerer Dateien zwischen einem Quell- und einem Ziel-Rechner. Daher ist eine direkte manuelle Nutzung inklusive Bearbeitung der Eingabe-Datei zu umständlich, was durch entsprechende Skripte abgefangen werden muss.

Das GT4 bietet die Möglichkeit an, Jobs an das Batch-System Condor zu schicken. Um diese Möglichkeit zu nutzen, muss bei der Installation die Source-Version genommen werden und eine entsprechende Option (--enable-wsgram-condor*)* aktiviert werden. Bei ersten Tests konnten zwar Jobs abgeschickt werden, diese wurden aber nur auf dem lokalen (GT4-)System ausgeführt, insofern dieses im Condor-Pool eingetragen war. Eine nähere Betrachtung der Vorgänge beim Einreichen eines Jobs über das GT4 an Condor hat ergeben, dass die Fehlerursache in dem in Perl geschriebenen Condor-Adapter lag, der bestimmte Condor-Optionen falsch gesetzt hatte. Der Condor-Adapter wurde entsprechend modifiziert. Im Verlauf des Projektes wurde ebenfalls der PBS-Adapter für TORQUE angepasst, weil dieser für die Ausführung eines Jobs gewisse Unzulänglichkeiten besaß. So wurde durch eine mangelhafte Implementierung beispielsweise vor jeder Job-Ausführung ein ssh-Login (wurde in WISENT durch rsh ersetzt) auf den Zielknoten gemacht. GT4 bietet über die Option „count“ die Möglichkeit an, den selben Jobs mehrmals zu starten. Das wird über ein Wrapper im Job-Adapter gesteuert, der eine Knotenliste bekommt und den Job dann der ssh-Login auf den entsprechenden Zielknoten startet. Das macht aus zwei Gründen keinen Sinn: 1) Bei count=1 wird immer ein ssh-Login auf den lokalen Knoten gemacht, der bereits vom Batch-System als Zielknoten ausgewählt wurde. Das führte u.a. dazu, dass Jobs nicht richtig suspendiert werden konnten, d.h. die Prozesse liefen weiter und der Jobs-Zustand war „suspend“. Im DGI führte dies ebenfalls zu Seiteneffekten, denn im Accounting konnte dadurch die CPUTime eines Jobs nicht richtig erfasst werden. Der Adapter wurde daher entsprechend für count=1 geändert. 2) Bei mehrfacher Ausführung des gleichen Jobs sollte das

Batch-System das Starten jedes einzelnen Jobs übernehmen, und nicht der Job-Adapter selbst. Letztlich war durch das einfache, auf Perl-Skripten basierende Job-Adapter-Konzept die Möglichkeiten gegeben, sehr schnell und einfach eigene Anpassungen vorzunehmen.

Insgesamt kann das GT4 als eine einsatzfähige Grid Middleware bezeichnet werden, die nutzbare Basisdienste anbietet. Dennoch ist das GT4 dem Namen nach wie ein „Toolkit“ zu verstehen, auf dem eigene, komplexere Dienste aufgebaut werden können. Insbesondere sind erweiterte Datenmanagementdienste wie der RLS (Replication Location Service) und der DRS (Data Replication Service) direkt nicht wirklich nutzbar. Die Benutzerfreundlichkeit der angebotenen Kommandozeilenwerkzeuge steht ohnehin nicht im Vordergrund, was für den Endnutzer zumindest über einfache Skripte abgefangen werden sollte. Die Entwicklung eigener Dienste bzw. Grid Services ist mit dem GT4 allerdings noch sehr komplex und es fehlt an ausgereifter Werkzeugunterstützung. Eine weitere Schwäche von GT4 ist auch die PerformanzPer. Es wurden dahingehend zwar keine umfangreichen Tests durchgeführt, aber der Betrieb des Compute-Clusters hat gezeigt, das insbesondere bei Einreichung mehrerer Jobs (>100) die Performanz stark leidet und GT4 viel Hauptspeicher (z.Z. 2 GB) zur Verfügung gestellt werden muss. Mittlerweile wurde GT4.2 veröffentlicht, was einige Neuerungen beinhaltet. Diese wurde aber im Rahmen von WISENT nicht betrachtet.

3.2 UNICORE 5

UNICORE steht für „Uniform Interface to Computing Resources“ und startete 1997 als ein BMBF-gefördertes Projekt und wurde später auch auf EU-Ebene gefördert. UNICORE hat sich besonders im Bereich der Hochleistungsrechenzentren verbreitet. Weil UNICORE bzw. UNICORE 5 eine unterstützte Grid-Middleware in D-Grid ist, wurde UNICORE 5 auch auf dem Compute-Cluster im OFFIS installiert. Vorher wurden bereits kleine Tests mit UNICORE 5 auf Testsystemen durchgeführt, um dieses mit dem GT4 zu vergleichen.

UNICORE 5 bietet als Einstieg ein Quickstart-Bundle an, mit dem schnell und einfach eine lauffähige UNICORE 5-Installation durchgeführt werden kann. Danach können Jobs allerdings nur auf dem lokalen System gestartet werden, eine Anbindung an ein Batch-System fehlt noch. Eine initiale, separate Installation und Einrichtung der einzelnen UNICORE-Komponenten (Gateway, NJS, TSI, UDDB) war sehr schwer durchzuführen, weil dazu genaue Kenntnisse der einzelnen Konfigurationseinstellungen notwendig sind. Es gibt aber keine zusammenhängende Dokumentation, sondern lediglich Einzeldokumente zu den einzelnen Komponenten. Dies erschwert einen manuelle Konfiguration mit geringen Vorkenntnissen.

Das DGI bietet ein modifiziertes Quickstart-Bundle sowie eine Installationsanleitung an, mit dem UNICORE 5 schnell und einfach auf dem Compute-Cluster installiert werden konnte. Ein kleiner Fehler, der sich im Installationsskript verbarg, wurde an die entsprechenden UNICORE-Ansprechpartner berichtet. Mit Hilfe dieser Ansprechpartner wurden auch kleinere Konfigurationseinstellungen überarbeitet und die UNICORE 5-Installation für D-Grid-Benutzer zugreifbar gemacht.

Ein Vorteil von UNICORE 5 ist der grafische Client, mit dem Jobs erstellt und an eine Compute-Ressource geschickt werden können. Zudem können einfache Workflows erstellt werden. Nachteilig erschien aber die notwendige manuelle Zuteilung eines Jobs an eine bestimmte Compute-Ressource. Eine Art Scheduling fehlt an dieser Stelle noch. Besonders umständlich ist dies bei größeren Workflows, bei dem jeder Teilschritt ein Job ist und entsprechend eine Compute-Ressource manuell zugeordnet werden muss. UNICORE 5 verfolgt allerdings eher den Ansatz, das der Nutzer sich selbst seine präferierte Ressource aussucht.

Ein großer Vorteil von UNICORE 5 ist, dass der Gateway als Schnittstelle nach außen einen einzigen Port benötigt, was sehr Firewall-freundlich ist. Dies bringt aber zugleich einen sehr großen Nachteil mit sich. Denn alle Datentransfers müssen zum einen durch den Gateway geleitet werden, zum anderen wird ein eigenes Datenübertragungsprotokoll verwendet. Die maximal erreichbare Durchsatzrate liegt bei ca. 200KB/s, was zur Übertragung der Datenmengen im WISENT-Kontext nicht geeignet ist. Allerdings geht UNICORE 5 eher davon aus, dass sich benötigte Daten schon auf der Compute-Ressource befinden, die vorher über andere, schnellere Datentransferprotokolle dorthin übertragen wurden.

Insgesamt ist UNICORE 5 die Grid-Middleware, die am schnellsten konfiguriert und installiert werden konnte. Allerdings hat UNICORE 5 noch einige Schwächen bzgl. der Unterstützung von Standards (es werden keinerlei Standards aus dem Grid-Umfeld unterstützt) und schneller Datentransfers. Der Nachfolger UNICORE 6 ist eine komplette Neuimplementierung der UNICORE 5-Komponenten in Java und bringt einige wesentliche Neuerungen mit sich. Eine der wichtigsten Neuerungen ist die Unterstützung von etlichen Standards wie WSRF, JSDL, usw., die zum Teil noch nicht vom GT4 (in den Versionen 4.0.x) unterstützt werden. UNICORE 6 kann generell als Vorreiter in der Aufnahme von Grid-relevanten Standars bezeichnet werden und hat damit einen großen Schritt in Richtung Interoperabilität vollzogen. Zusätzlich hat UNICORE 6 sowohl Client- als auch Server-seitig eine ausgereifte Plugin-Funktionalität, um eigene Erweiterungen einzubringen. UNICORE 6 hat sich grundlegend weiter entwickelt und wird sich voraussichtlich als Grid-Middleware in Zukunft stärker etablieren können.

3.3 gLite

gLite ist eine sehr umfangreiche Grid-Middleware, die im Projekt EGEE (Enabling Grids for E-SciencE) unter der besonderen Berücksichtigung von Anforderungen der Hochenergiephysikanwendungen entwickelt wurde. Aufgrund der Größe des EGEE-Projekts ist gLite die am weitesten verbreitete produktiv eingesetzte Grid-Middleware überhaupt. Im Rahmen der vom BMBF finanzierten D-Grid-Sonderinvestition wird gLite 3.0 neben GT4 und UNICORE als Zugangsmechanismus für die angeschafften Cluster verwendet. Für WISENT bietet die Auseinandersetzung mit gLite eine Möglichkeit der zukünftigen Ressourcennutzung über das D-Grid hinaus.

Die Installation von gLite wurde mit Hilfe einer durch das DGI-Projekt erstellten Anleitung und eines vorgefertigten Festplattenimage vorgenommen. Trotz dieser Hilfestellungen war die Installation von gLite im Vergleich zu den beiden anderen Middleware-Paketen bei weitem die aufwändigste.

gLite 3.0 ist nur eingeschränkt portabel zwischen verschiedenen Linux-Versionen und erfordert die Verwendung von Scientific Linux 3.0.8, einer beim CERN speziell für das Grid entwickelten Linux-Distribution. Da die zentrale IT-Administration in OFFIS ausschließlich SuSE Linux unterstützt (insb. was die Aktualisierung mit Patches betrifft), und da der Cluster in einer Single-System-Image-Konfiguration betrieben wird (die Software wird auf dem Master-Knoten installiert und ist dann automatisch auf allen Rechenknoten verfügbar), wurde im gesamten Cluster SuSE Linux Enterprise 10 installiert. Aus diesem Grund musste gLite in einer virtuellen Maschine (Xen), ähnlich wie in der Referenzinstallation des DGI, untergebracht werden.

Die ersten Probleme ergaben sich bei der Xen-Installation. Die vom DGI gelieferte Anleitung erklärte nicht die Details der notwendigen Netzwerk- und Routingkonfiguration für die zwei installierten Netzwerkkarten und die Interaktion von Einstellungen mit der Firewall auf dem Master-Knoten. Zudem musste zum Hochfahren der virtuellen Maschine eine neue Ram-Disk erstellt werden, was ebenfalls erst im Laufe von gescheiterten Versuchen herausgefunden werden konnte.

Nach der erfolgreichen Einrichtung der virtuellen Maschine musste die gLite-Konfiguration und die Installation von zusätzlichen auf dem Festplattenimage nicht mitgelieferten Modulen durchgeführt werden. Dabei wurden einige Unklarheiten in der Anleitung des DGI entdeckt, die von den DGI-Ansprechpartnern beseitigt und in eine aktualisierte Version der Anleitung eingebracht wurden. Die Tests von bereits installierten Paketen wurden dadurch erschwert, dass die für gLite notwendige zentrale Infrastruktur in Karlsruhe zeitgleich mit der WISENT-Installation aufgesetzt wurde. Deswegen war bei manchen aufgetretenen Problemen nicht sofort klar, ob ihre Ursache auf der WISENT- oder DGI-Seite lag.

Im Rahmen der Installation wurde ein Fehler in einem der Job-Manager von gLite entdeckt, der die Berechnung des freien Arbeitsspeichers auf dem Master-Knoten betraf und zur Ablehnung von abgeschickten Jobs führte. Dieser Fehler musste im vorliegenden Perl-Code manuell korrigiert werden.

Es war während der Installation nicht klar, ob gLite mit der auf den Rechenknoten installierten Version von SuSE Enterprise Linux überhaupt zusammenarbeiten kann. In der DGI-Referenzinstallation wurde stattdessen Scientific Linux für die Knoten verwendet, was in OFFIS aus den oben genannten Gründen nicht durchgesetzt wurde. Durch die erfolgreiche Einrichtung der gewünschten Konfiguration in WISENT wurden Erfahrungen gesammelt, die an DGI berichtet wurden und in die gemeinsame Installationsanleitung einflossen.

Eine Reihe von Problemen wurde durch die Ausschaltung des Job-Managers „fork“ verursacht. Dieser Job-Manager ermöglicht Grid-Nutzern die Ausführung von beliebigen Programmen auf dem Master-Knoten, was generell als Sicherheitslücke einzustufen ist. Trotzdem muss er bei gLite 3.0 eingeschaltet bleiben, wie unsere Erfahrungen zeigten. Es fiel auf, dass während der gescheiterten Job-Ausführung Fehlermeldungen sichtbar wurden, die allerdings nichts mit dem ausgeschalteten Job-Manager zu tun hatten und alleine die Job-Ausführung nicht beeinträchtigten. Trotzdem wurde sehr viel Zeit in die genaue Diagnose der Ursachen dieser Fehler investiert, die nicht nur Eingriffe in verschiedene von gLite verwendete Skripte, sondern auch das Herunterladen und die Übersetzung des gLite-Quellcode

erforderte (zum Debugging). Zusammenfassend litt der gesamte gLite-Installationsprozess darunter, dass das beobachtete Verhalten der Software nur mühsam in korrekt, anscheinend korrekt, falsch oder anscheinend falsch klassifiziert werden konnte. Die Verständlichkeit von jeglichen Fehlermeldungen, die von gLite berichtet wurden, war sehr schlecht.

Trotz Schwierigkeiten bei der Installation war WISENT eins der ersten mit der Sonderinvestition finanzierten Projekte, die den Zugriff auf die angeschafften Ressourcen mit gLite gemäß Anforderungen des DGI ermöglichten. Nach der erfolgreichen Einrichtung von gLite 3.0 wurden Tests mit verschiedenen Jobs durchgeführt, die sowohl an den lokalen Cluster in OFFIS als auch an andere bereits mit gLite erreichbare D-Grid-Cluster abgeschickt wurden. Unter anderem wurden erfolgreich virtuelle Maschinen (realisiert mit User Mode Linux) zum Rechenzentrum in Hannover verschickt, um dort in einer von WISENT festgelegten Softwareumgebung einige Testprogramme auszuführen. Beim Abschicken von gLite-Jobs fiel negativ auf, dass der gesamte Vorgang sehr langsam ist – es verliefen stets mehrere Minuten von der Job-Einreichung bis zur Ausführung bzw. dem Einreichen des Jobs an ein Batch-System. Die Zeitverzögerung beteiligter Batch-Systeme hat auf Grund hinreichend verfügbarer Ressourcen keine wesentliche Rolle gespielt.

4.Datenmanagement

Neben dem Zugriff den Rechenressourcen und der Ausführung von Jobs spielt das Datenmanagement für Input bzw. Output-Daten eine zentrale Rolle. Zum Teil wurde dies schon in Abschnitt 3 bei der jeweiligen Grid-Middleware erwähnt. Dieser Abschnitt konzentriert sich auf das in WISENT produktiv eingesetzte GridFTP als Datentransferprotokoll (Abschnitt 4.1) sowie dCache als Schnittstelle zu großen Speicherressourcen (Abschnitt 4.2).

4.1 GridFTP

Neben der Durchführung von Datentransfertest (Abschnitt 4.1.1), werden in diesem Abschnitt zwei Einsatzszenarien des GridFTP vorgestellt, in denen sicherheitsrelevante Problemstellungen fokussiert wurden. Dazu gehört der Umgebung mit strengen Sicherheitsrichtlinien und daraus resultierenden strengen Firewalls bei DLR DFD (Abschnitt 4.1.2) sowie die Einrichtung eines GridFTP-Servers in einer „chroot“-Umgebung an der Universität Oldenburg und dessen Einbindung in einen komplexeren Workflow mit Nutzung von Rechenressourcen und einer dCache-Ressource des D-Grid (Abschnitt 4.1.3).

4.1.1 Datendurchsatztests

GridFTP bietet die Option an, für einen Datentransfer mehrere Datenkanäle gleichzeitig zu nutzen. Um den Nutzen dieser Option zu bestimmen, wurden kleinere Tests durchgeführt. An den Testszenarien waren drei GridFTP-Server beteiligt:

- srvgrid.offis.uni-oldenburg.de - OFFIS
- eridanus.caf.dlr.de - DLR

- juggle-glob.fz-juelich.de - Jülich

Als Testdatei wurde eine 1GB-Datei (1073741824 Bytes) verwendet. In einem Durchgang wurde die Datei 10 mal übertragen. Der erste Transfer in einem Durchgang begann mit einem Datenkanal, für jeden weiteren Transfer wurde die Anzahl um 1 inkrementiert. Der 10te Transfer in einem Durchgang hat demnach 10 Datenkanäle genutzt. Jeder Durchgang wurde für jeden möglichen Transferweg (DLR->Jülich, Jülich->DLR, OFFIS->DLR, DLR->OFFIS, OFFIS->Jülich, Jülich->OFFIS) 3 mal durchgeführt. Dabei wurden die benötigte Zeit für einen Transfer sowie die von GridFTP gemessene mittlere Durchsatzrate protokolliert und am Ende pro Transferweg ein Mittelwert gebildet. Der anschließend errechnete „Speedup“ gibt den Beschleunigungsfaktor im Verhältnis zu einem Datenkanal an. Ein paar Werte wurden als Ausreißer aus der Wertung genommen (vermutlich eine kurzzeitig sehr stark ausgelastete Leitung), weil sie sonst die Mittelwerte zu stark beeinflusst hätten. Im Prinzip können in der Realität insbesondere durch ausgelastet Leitungen immer Abweichungen von den (unter „optimalen“ Bedingungen gemessenen) Mittelwerten auftreten. Von den insgesamt 180 durchgeführten Transfers schlugen zwei fehl. Alle Transfers waren so genannte Third-Party-Transfers, die von einem OFFIS-Rechner aus initiiert wurden.

In einem weiteren Test wurden die Transferwege OFFIS->DLR, DLR->OFFIS, OFFIS->Jülich, Jülich->OFFIS nochmals durchlaufen, weil der OFFIS-Cluster eine Gigabit-Anbindung bekommen hatte (vorher waren es 100MBit).

Auf weitere Tests wurde verzichtet, weil auf Basis der Ergebnisse die Performanz von GridFTP gut eingeschätzt werden konnte. Im Wesentlichen ging es bei diesem Test darum, herauszufinden, welche Übertragungsraten die einzelnen Verbindungswege ermöglichen und welche Anzahl paralleler Datenkanäle dabei empfohlen werden kann. Weitere Tests könnten z.B. die Übertragung kleinerer Dateien beinhalten oder die Nutzung von mehreren Daten-Knoten.

Die folgenden Tabellen enthalten die protokollierten Durchsätze und Beschleunigungsfaktoren in Abhängigkeit von der Anzahl der Datenkanäle. In den angefügten Abbildungen ist die Abhängigkeit zwischen der Anzahl der Datenkanäle und dem (mittleren) Durchsatz dargestellt bzw. der Vergleich von Upload und Download für eine Verbindung.

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	107,108	9,78	105,278	9,93	109,867	9,5	107,42	9,74	
2	79,298	13,3	82,927	12,67	90,231	11,68	84,15	12,55	1,29
3	72,76	14,52	66,217	16,03	68,345	15,44	69,11	15,33	1,57
4	60,525	17,56	66,46	15,98	70,836	14,97	65,94	16,17	1,66
5	60,356	17,59	64,176	16,6	68,866	15,38	64,47	16,52	1,7
6	61,017	17,44	74,592	14,22	81,568	12,95	72,39	14,87	1.53
7	59,457	17,87	62,702	16,98	68,574	15,47	63,58	16,77	1,72
8	60,024	17,72	62,166	17,1	76,225	13,8	66,14	16,21	1,66
9	60,942	17,44	86,539	12,15	64,814	16,41	70,77	15,33	1,57
10	66,934	15,85	55,315	19,39	59,887	17,81	60,71	17,68	1,82

Tabelle 1: Datenübertragungstest DLR->Jülich am 02.08.2007

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	82,244	12,74	74,61	15,13	77,803	14,42	78,22	14,1	
2	56,642	19,07	55,501	20,28	53,676	20,77	55,27	20,04	1,42
3	49,025	22,6	42,529	26,39	44,12	26,26	45,22	25,08	1,78
4	43,008	26,39	43,282	26,6	42,41	27,38	42,9	26,79	1,9
5	40,904	29,26	40,197	28,6	48,99	24,91	43,36	27,59	1,96
6	48,979	23,98	40,998	28,76	40,733	29,2	43,57	27,31	1,94
7	45,071	26,67	41,489	29,17	39,203	29,26	41,92	28,37	2,01
8	41,906	29,17	43,895	27,9	38,999	29,68	41,6	28,92	2,05
9	40,006	29,26	42,726	29,34	41,265	29,68	41,33	29,43	2,09
10	41,482	28,37	44,569	28,21	42,956	26,06	43	27,55	1,95

Tabelle 2: Datenübertragungstest Jülich->DLR am 02.08.2007

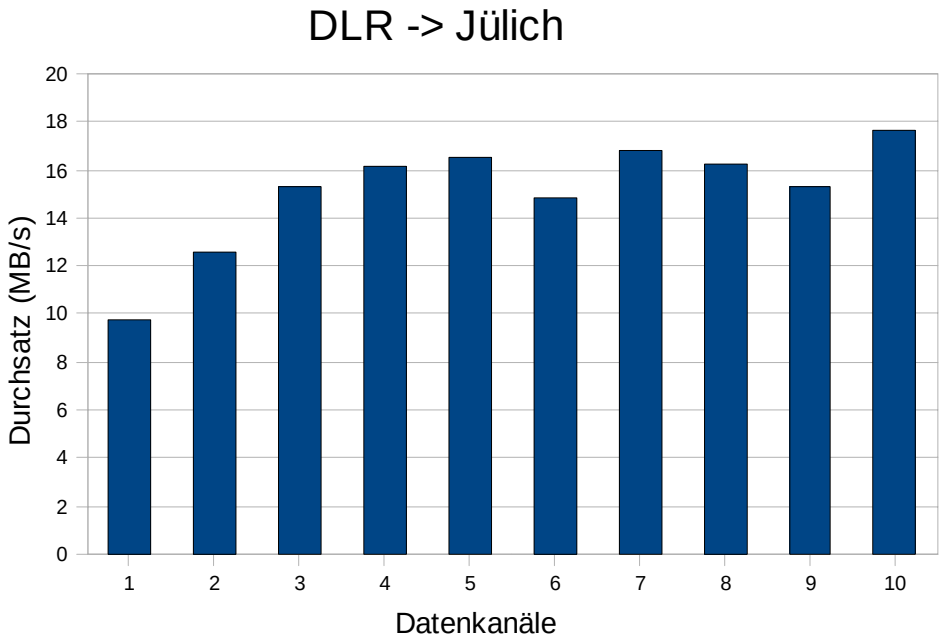


Abbildung 1: Datenübertragungstest DLR->Jülich

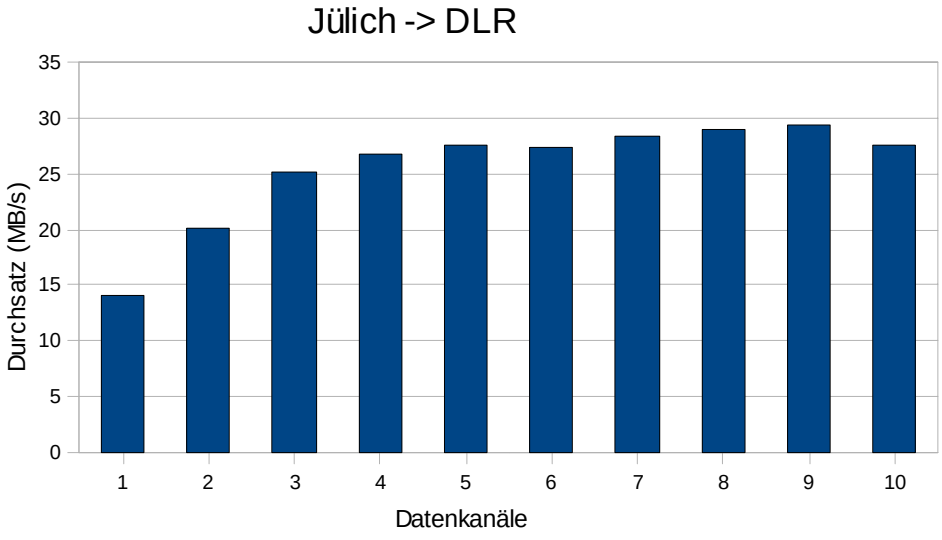


Abbildung 2: Datenübertragungstest Jülich->DLR

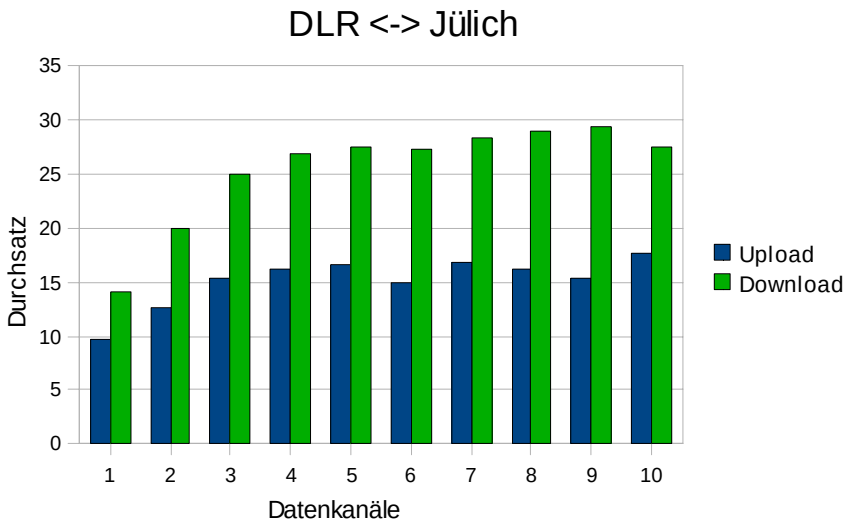


Abbildung 3: Vergleich Upload/Download

Anzahl Daten-kanäle	Durch-gang 1 Zeit (s)	Durch-gang 1 Durch-satz (MB/s)	Durch-gang 2 Zeit (s)	Durch-gang 2 Durch-satz (MB/s)	Durch-gang 3 Zeit (s)	Durch-gang 3 Durch-satz (MB/s)	Durch-schnitt Zeit (s)	Durch-schnitt Durch-satz (MB/s)	Speed up Durch satz
1	414,693	2,48	426,513	2,41	418,028	2,47	419,74	2,45	
2	208,291	4,98	200,965	4,69	213,19	4,85	207,48	4,84	1,98
3	153,392	6,87	169,916	6,16	152,562	6,81	158,62	6,61	2,7
4	119,971	8,74	125,537	8,33	125,821	8,28	123,78	8,45	3,45
5	107,276	9,87	106,953	9,89	105,083	9,95	106,44	9,9	4,04
6	101,868	10,26	103,926	10,13	106,411	9,83	104,07	10,07	4,11
7	101,016	10,49	100,311	10,51	101,422	10,32	100,92	10,44	4,26
8	97,158	10,88	97,004	10,89	100,771	10,5	98,31	10,76	4,39
9	97,654	10,82	98,561	10,66	99,563	10,64	98,59	10,71	4,37
10	98,943	10,73	100,569	10,58	98,645	10,63	99,39	10,65	4,35

Tabelle 3: Datenübertragungstest OFFIS->DLR am 03.08.2007

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	897,476	1,15	590,05	1,75	529,404	1,88	672,31	1,59	
2	842,815	1,22	356,942	2,89	307,54	3,55	502,43	2,55	1,6
3	511,44	2,02	301,986	3,43	227,851	4,66	347,09	3,37	2,12
4	206,284	5,03	205,436	5,05	184,214	5,89	198,64	5,32	3,35
5	168,031	6,19	154,58	6,9	145,245	7,05	155,95	6,71	4,22
6	145,782	7,24	161,604	6,42	148,075	7,35	151,82	7	4,4
7	137,786	7,87	127,025	8,22	129,839	8,49	131,55	8,19	5,15
8	123,659	8,52	117,562	9,1	116,774	8,87	119,33	8,83	5,55
9	111,17	9,43	108,635	9,73	104,012	10,73	107,94	9,96	6,26
10	107,375	9,82	107,158	9,93	108,013	10,81	107,52	10,19	6,41

Tabelle 4: Datenübertragungstest OFFIS->DLR am 21.02.2008, mit Gigabit Ethernet

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	1886,14	0,54	1740,87	0,59	1779,76	0,58	1802,26	0,57	
2	995,024	1,03	986,066	1,04	982,825	1,04	987,97	1,04	1,82
3	676,439	1,52	703,284	1,46	719,451	1,43	699,72	1,47	2,58
4	587,402	1,75	562,21	1,83	567,164	1,81	572,26	1,8	3,16
5	460,131	2,23	463,629	2,22	460,359	2,23	461,37	2,23	3,91
6	410,08	2,51	420,214	2,45	427,471	2,41	419,25	2,46	4,32
7	406,521	2,54	382,219	2,69	363,176	2,83	383,97	2,69	4,72
8	371,15	2,78	353,607	2,91	328,256	3,14	351	2,94	5,16
9	331,983	3,1	318,283	3,23	312,415	3,3	320,89	3,21	5,63
10	299,991	3,44	289,47	3,44	287,211	3,59	292,22	3,49	6,12

Tabelle 5: Datenübertragungstest DLR->OFFIS am 03.08.2007

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	448,703	2,32	462,906	2,23	461,927	1,48	457,85	2,01	
2	255,204	4,07	264,824	3,93	266,192	2,59	262,07	3,53	1,76
3	179,288	5,79	191,219	5,46	167,527	3,35	179,34	4,87	2,42
4	168,03	6,19	153,163	6,79	154,59	4,39	158,59	5,79	2,88
5	144,374	7,22	139,291	7,53	133,695	5,06	139,12	6,6	3,28
6	127,775	8,16	133,186	7,94	109,803	5,02	123,59	7,04	3,5
7	144,133	7,56	118,233	8,8	140,33	5,19	134,23	7,18	3,57
8	117,461	8,97	109,068	10,36	119,192	4,73	115,24	8,02	3,99
9	128,597	8,32	73,693	14,34	93,321	5,14	98,54	9,27	4,61
10	105,769	10,19	102,452	10,76	95,444	5,54	101,22	8,83	4,39

Tabelle 6: Datenübertragungstest DLR->OFFIS am 21.02.2008, mit Gigabit Ethernet

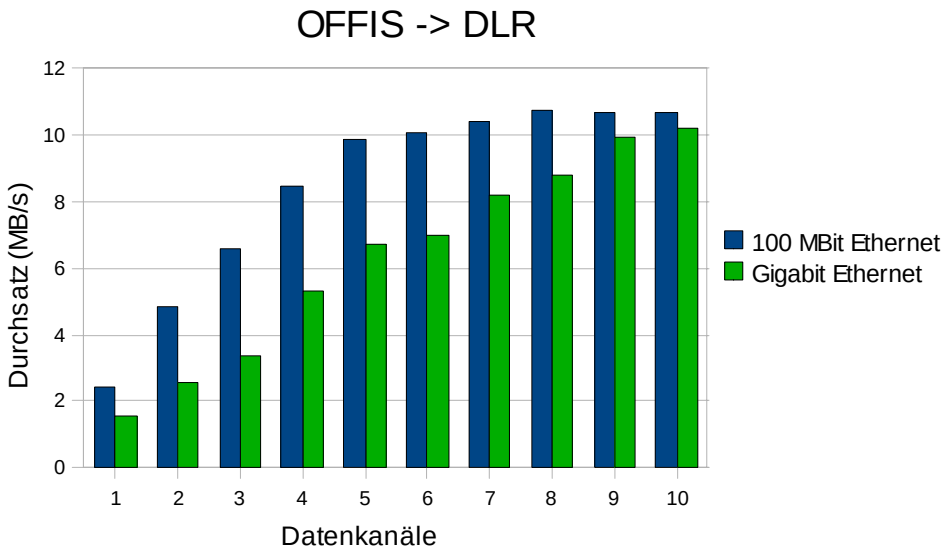
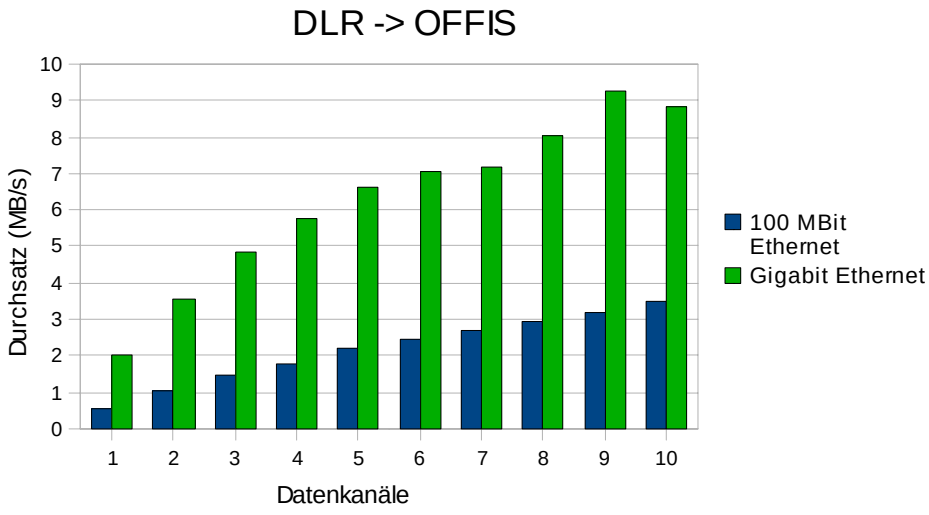


Abbildung 4: Datenübertragungstest OFFIS->DLR



A

bbildung 5: Datenübertragungstest OFFIS->DLR

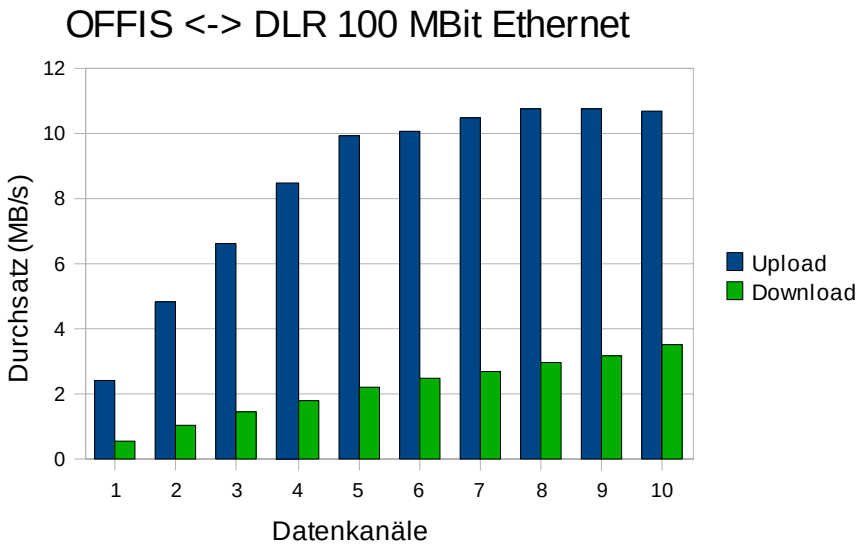


Abbildung 6: Vergleich Upload/Download 100 MBit Ethernet

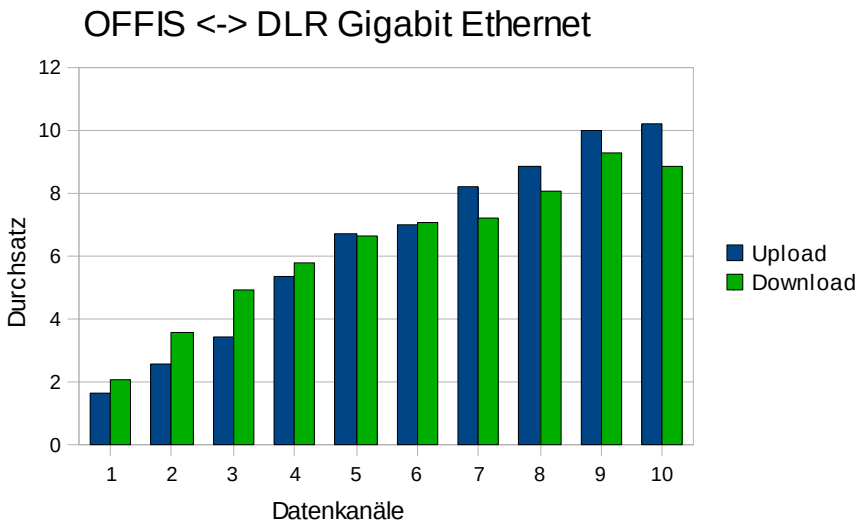


Abbildung 7: Vergleich Upload/Download Gigabit Ethernet

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	446,296	2,3	434,99	2,36	442,128	2,33	441,14	2,33	
2	226,008	4,56	228,887	4,5	373,66	2,76	276,19	3,94	1,69
3	158,425	6,52	156,937	6,58	160,031	6,45	158,46	6,52	2,8
4	124,734	8,29	123,341	8,39	127,227	8,15	125,1	8,28	3,55
5	**Failed**	**Failed**	108,677	9,55	112,029	9,25	110,35	9,4	4,03
6	111,791	9,72	108,445	9,56	109,455	9,47	109,9	9,58	4,11
7	106,664	9,72	109,725	9,46	108,977	9,52	108,46	9,57	4,11
8	106,752	9,72	108,137	9,59	110,85	9,37	108,58	9,56	4,1
9	106,757	9,72	*****175,89	*****5.87	110,985	9,35	108,87	9,54	4,09
10	106,571	9,76	*****268,32	*****3.87	108,667	9,55	107,62	9,66	4,15

*Tabelle 7: Datenübertragungstest OFFIS->Jülich am 06.08.2007 (***** Ausreisser, nicht gewertet)*

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	822,386	1,25	457,125	2,25	671,458	1,53	650,32	1,68	
2	697,589	1,47	237,427	4,35	366,055	2,81	433,69	2,88	1,71
3	364,469	2,85	163,045	6,36	166,192	6,22	231,24	5,14	3,06
4	129,088	8,02	129,763	7,98	132,473	7,82	130,44	7,94	4,73
5	104,57	9,92	105,042	9,88	107,107	9,81	105,57	9,87	5,88
6	94,246	11,02	89,973	11,56	91,867	11,31	92,03	11,3	6,73
7	89,869	11,57	79,711	13,08	82,529	12,63	84,04	12,43	7,4
8	*****223,50	*****4,67	71,094	14,69	77,124	13,54	74,11	14,11	8,4
9	*****343,34	*****3	66,811	15,68	70,496	14,84	68,65	15,26	9,08
10	63,27	16,57	62,762	16,7	62,144	16,87	62,73	16,71	9,95

*Tabelle 8: Datenübertragungstest OFFIS->Jülich am 20.02.2008, mit Gigabit Ethernet (***** Ausreisser, nicht gewertet)*

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	463,33	2,22	445,143	2,31	467,254	2,2	458,58	2,24	
2	237,029	4,35	234,111	4,4	229,5	4,49	233,55	4,41	1,97
3	166,346	6,21	168,003	6,15	180,857	5,71	171,74	6,02	2,69
4	129,069	8,03	131,497	7,87	136,327	7,59	132,3	7,83	3,5
5	135,961	7,61	122,801	8,43	137,604	7,52	132,12	7,85	3,5
6	129,553	7,99	123,378	8,4	117,461	8,83	123,46	8,41	3,75
7	119,487	8,68	123,024	8,43	118,32	8,76	120,28	8,62	3,85
8	117,113	8,86	119,175	8,71	116,631	8,9	117,64	8,82	3,94
9	113,821	9,1	113,365	9,14	**Failed**	**Failed**	113,59	9,12	4,07
10	110,783	9,37	*****149.19	*****6.93	115,808	9,18	113,3	9,27	4,14

Tabelle 9: Datenübertragungstest Jülich->OFFIS am 06.08.2007
*(***** Ausreisser, nicht gewertet)*

Anzahl Datenkanäle	Durchgang 1 Zeit (s)	Durchgang 1 Durchsatz (MB/s)	Durchgang 2 Zeit (s)	Durchgang 2 Durchsatz (MB/s)	Durchgang 3 Zeit (s)	Durchgang 3 Durchsatz (MB/s)	Durchschnitt Zeit (s)	Durchschnitt Durchsatz (MB/s)	Speed up Durchsatz
1	446,836	2,34	434,612	2,37	449,39	2,29	443,61	2,33	
2	223,014	4,63	220,878	4,67	216,157	4,78	220,02	4,69	2,01
3	144,763	7,16	150,254	6,91	145,512	7,12	146,84	7,06	3,03
4	113,933	9,44	118,562	8,81	112,103	9,53	114,87	9,26	3,97
5	86,626	12,08	89,005	11,8	88,374	12,08	88	11,99	5,15
6	82,56	14,01	83,164	13,91	75,333	13,95	80,35	13,96	5,99
7	67,382	16,25	71,739	16,28	78,519	14,95	72,55	15,83	6,79
8	68,084	18,89	56,47	18,75	82,207	19,25	68,92	18,96	8,14
9	51,021	21,38	55,869	21,16	85,103	16,9	64	19,81	8,5
10	45,945	23,22	66,997	23,12	89,702	23,49	67,55	23,28	9,99

Tabelle 10: Datenübertragungstest Jülich->OFFIS am 20.02.2008, mit Gigabit Ethernet

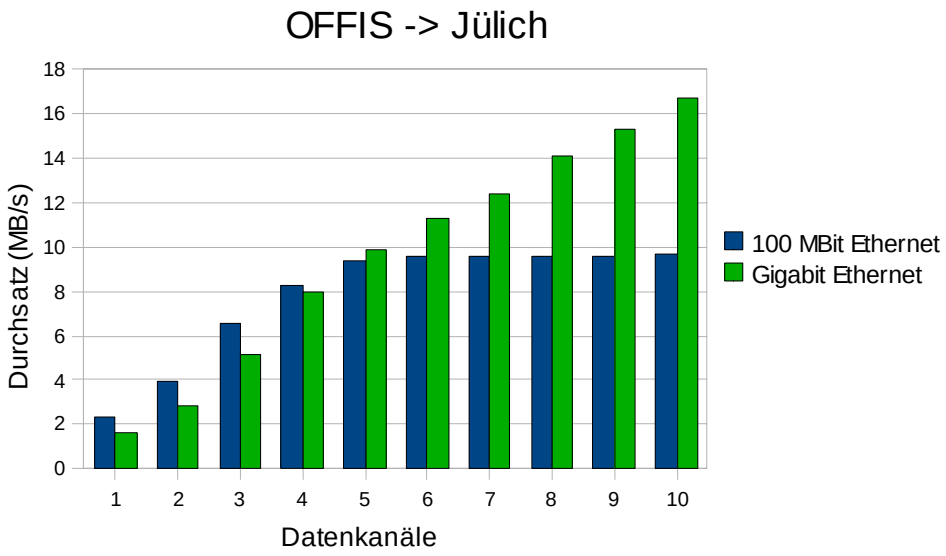


Abbildung 8: Datenübertragungstest OFFIS->Jülich

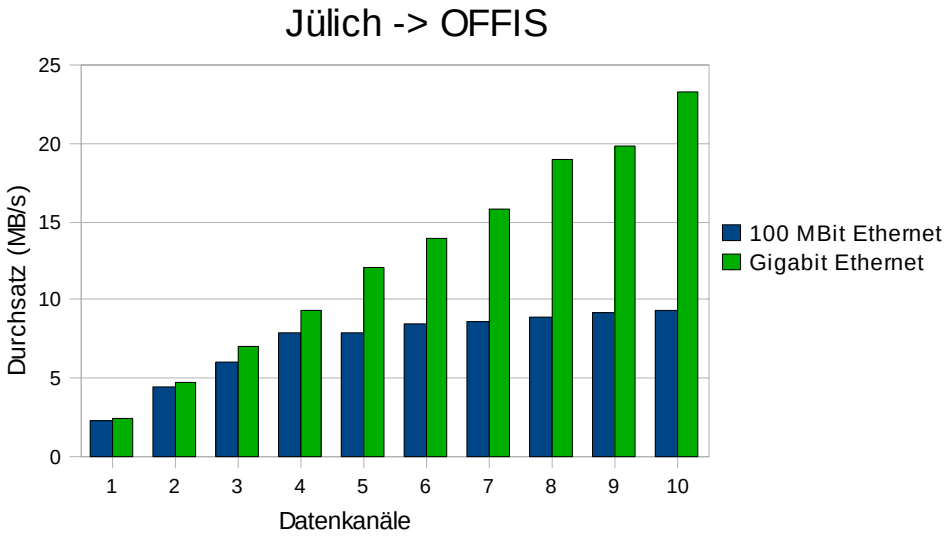


Abbildung 9: Datenübertragungstest Jülich->OFFIS

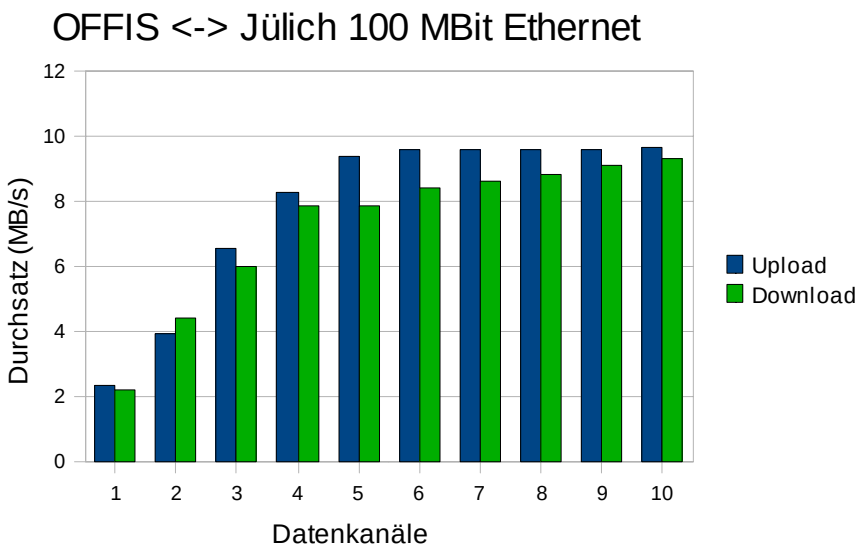


Abbildung 10: Vergleich Upload/Download 100 MBit Ethernet

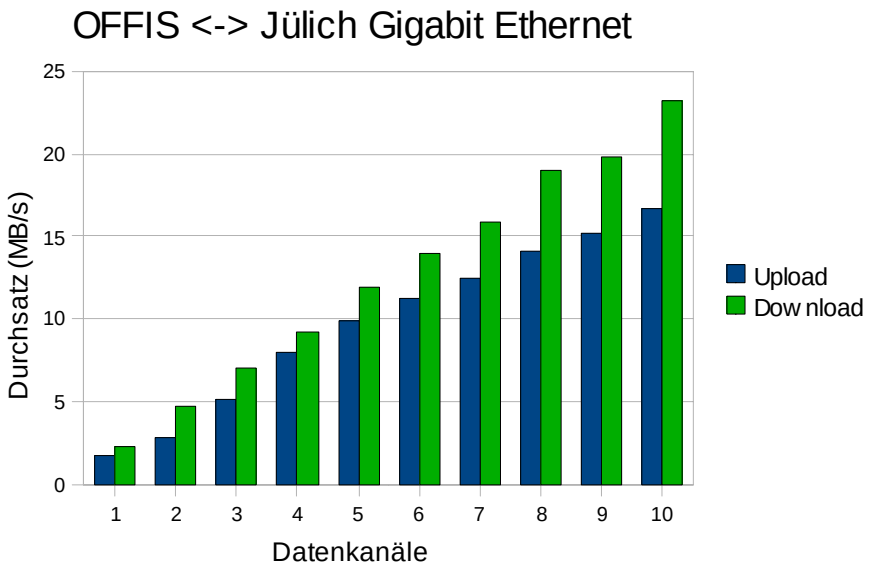


Abbildung 11: Vergleich Upload/Download Gigabit Ethernet

An den Tabellen und insbesondere an den Abbildungen ist zu erkennen, dass jeder Transferweg ein unterschiedliches, aber eindeutiges Bild zeigt.

- DLR<->Jülich: Bei beiden Richtungen ist zu erkennen, dass ein einzelner Datenkanal schon einen guten Durchsatz erreicht, ca. 10 MB/s bei der Richtung DLR->Jülich (siehe Abbildung 1) und ca. 14 MB/s bei der Richtung Jülich->DLR (siehe Abbildung 2). Bei weiteren Datenkanälen zeigt sich, dass der Gesamtdurchsatz nur mäßig steigt und ab ca. 5 parallelen Datenkanälen keine wesentliche Steigerung (zum Teil schwankend und rückläufig) mehr zu erkennen ist. In Abbildung 3 ist zu sehen, dass die Richtung Jülich->DLR deutlich schneller ist als die Richtung DLR->Jülich.
- OFFIS<->DLR: Diese Verbindung zeigt es etwas differenziertes Bild, was u.a. mit der unterschiedlichen Auslastung der Internet-Verbindung beispielsweise beim DLR zusammenhängen kann. Bei der Richtung OFFIS->DLR (siehe Abbildung 4) ist zu erkennen, dass bei der 100MBit-Leitung die Durchsatzrate schneller ansteigt als bei der Gigabit-Leitung, bei 10 parallelen Datenkanälen aber bei beiden Leitungen ungefähr derselbe Durchsatz von ca. 10 MB/s erreicht wird. Es ist zu vermuten, dass bei der Gigabit-Leitung noch mehr Durchsatz bei mehr als 10 Datenkanälen zu erreichen ist, bei der 100MBit-Leitung ist bei 5 parallelen Datenkanälen die physikalische Grenze (theoretisch 12,5 MB/s) der Leitung erreicht. Bei der anderen Richtung DLR->OFFIS (siehe Abbildung 5) scheint beim Test mit der 100MBit-Leitung ein Teil der

Verbindungsstrecke (vermutlich beim DLR) ausgelastet gewesen zu ein oder es gab zu dem Zeitpunkt eine unterschiedliche Strategie bzgl. der erlaubten Durchsatzes beim Upload pro Verbindung. Die Gigabit-Leitung ist jedenfalls deutlich schneller und kann vermutlich bei mehr als 10 parallelen Datenkanälen noch zulegen. Die 100MBit-Leitung hat dann bei ca. 10 MB/s ihr physikalisches Maximum erreicht. In der Abbildung 6 ist nochmal zu sehen, dass bei der 100MBit-Leitung die Richtung OFFIS->DLR deutlich schneller ist als die Richtung DLR->OFFIS. Bei der Gigabit-Leitung sind beide Richtungen ziemlich ausgeglichen, siehe Abbildung 7.

- OFFIS<->Jülich: Bei dieser Verbindung ist ein klares Bild zu erkennen. In beiden Richtungen (siehe Abbildungen 8 und 9) ist ab ca. 4-5 parallelen Datenkanälen die physikalische Grenze bei der 100MBit-Leitung erreicht. Bei der Gigabit-Leitung gibt es bis 10 parallelen Datenkanälen einen nahezu linearen Anstieg des Durchsatzes, der Durchsatz skaliert somit sehr gut. Bei weiteren kurzen Tests konnte mit mehr als 10 parallelen Datenkanälen ein Durchsatz von über 30MB/s erreicht werden, dort liegt auch die Grenze. In Abbildung 10 ist zu sehen, dass bei der 100MBit-Leitung die Richtung OFFIS->Jülich etwas schneller ist als die Richtung Jülich->OFFIS, aber im Prinzip befanden sich beide Richtungen an der physikalischen 100MBit-Grenze, ggf. war bei der Richtung Jülich->OFFIS die Leitung schon etwas ausgelastet. In Abbildung 11 ist zu hingegen zu erkennen, dass die Richtung Jülich->OFFIS schneller ist als die Richtung OFFIS->Jülich.

Abschließend kann festgehalten werden, dass mit fünf parallelen Datenkanälen für eigentliche alle Richtungen „optimale“ Durchsatzraten erreicht werden können, je nachdem, was die Leitung hergibt. Es hat sich gezeigt, dass lange Zeit die 100MBit-Anbindung des Clusters beim OFFIS ein Flaschenhals war und nach Aufrüstung eines Gigabit-Anschlusses zumindest mit Jülich deutliche bessere Ergebnisse erzielt werden konnten. Das OFFIS selbst ist über die Universität Oldenburg mit einem Gigabit-Anschluss mit dem DFN verbunden.

4.1.2 Erfahrungen bei DLR DFD

Auf Seiten des DLR-DFD bestand die größte Herausforderung bei der Einrichtung eines GridFTP-Server in dem Umgang mit den strengen Firewall-Richtlinien. Dies wurde bereits ausführlich in dem Dokument „AP 2.1 - Bericht zu den Auswirkungen der DLR-IT-Sicherheitsstrukturen beim Datentransfer mit Partnern außerhalb des DLR“ beschrieben. Zusammenfassend ist die Situation so, dass es internes DLR-Netz gibt und daneben eine DMZ (demilitarized zone – entmilitarisierte Zone). Verbindungen von außen in das interne DLR-Netz sind gänzlich verboten und vom DLR-Netz nach außen ebenfalls stark eingeschränkt. Die DLR-DMZ ist typischerweise etwas lockerer, d.h. es werden auch Verbindungen von außen zugelassen. Verbindungen von der DLR-DMZ in das interne DLR-Netzwerk sind wiederum sehr restriktiv bzw. eigentlich verboten. Im Endeffekt bedeutet dies, dass für die bidirektionale Kommunikation nach außen die DLR-DMZ zuständig ist und nach Möglichkeit Verbindungen zwischen DLR-DMZ und dem internen DLR-Netz vom internen DLR-Netz aus initiiert werden sollten, um die DLR-DMZ gut vom internen DLR-Netz zu isolieren. Prinzipiell gesehen sind auch Ausnahmen möglich, aber die Anforderungen von GridFTP an geöffneten Ports lässt nur ein Einsatz in

der DLR-DMZ zu. Neben dem statischen Port für Kontrollverbindungen 2811 werden mehrere dynamische Ports für Datenverbindungen benötigt. Dieser Bereich kann zwar eingeschränkt werden, im D-Grid wird der Bereich 20000-25000 empfohlen, ist aber nur mit der DLR-DMZ verträglich. Um dennoch Daten mit dem internen DLR-Netz austauschen zu können, wurde zwischen dem Rechner `eridanus` in der DLR-DMZ und einer Speichereinheit (beides D-Grid Sonderinvestition) im internen DLR-Netz ein Synchronisationsmechanismus (`rsync`) eingerichtet, der periodisch initiiert aus dem DLR-Netz heraus einen Bereich des Dateisystems von `eridanus` auf die Speichereinheit synchronisiert und dies für einen weiteren Bereich genau umgekehrt durchführt. So können zwischen DLR-DMZ und dem internen DLR-Netz bidirektional Daten ausgetauscht werden. Diese Technik wurde u.a. auch zur Umsetzung des MSG-Tailor-Web Services verwendet, siehe Dokument „AP3.3 - Einsatz von und Erfahrungen mit Web Service-Technologien“.

Die Installation und Einrichtung von Globus Toolkit 4 bzw. GridFTP auf `eridanus` konnte nach der üblichen Anleitung durchgeführt werden, nachdem zuvor Betriebssystem-bedingte Problem mit Sun Solaris 9 behoben wurde, siehe Abschnitt 3.1. Lediglich bei der Konfiguration musste darauf geachtet werden, dass die beiden Netzwerkkarten richtig unterstützt werden. Die erste Netzwerkkarte hat eine interne IP-Adresse und ist nur von dem internen DLR-Netz erreichbar. Die zweite Netzwerkkarte hat eine interne IP-Adresse für die DLR-DMZ, die aber nach außen hin einer globalen IP-Adresse zugeordnet wird. Daher werden auf `eridanus` zwei GridFTP-Server gestartet, jeweils einer für jede Netzwerkkarte. Für die externe Netzwerkkarte wurde der GridFTP-Server so konfiguriert, dass die externe IP-Adresse verwendet wird. Für die interne Netzwerkkarte die entsprechende interne IP-Adresse. Danach war der „externe“ GridFTP-Server einsatzbereit und hat alle funktionalen Tests bestanden. Es wurden auch verschiedene Durchsatzraten getestet, siehe Abschnitt 4.1.1. Der „interne“ GridFTP-Server wurde nicht getestet, weil der Synchronisationsmechanismus zum Datenaustausch mit dem internen DLR-Netz zur Zeit ausreichend ist. Prinzipiell sollte die Richtung internes DLR-Netzwerk → DLR-DMZ funktionieren, die umgekehrte Richtung ist aus den bereits erwähnten Firewall-technischen Einschränkungen nicht möglich.

4.1.3 Erfahrungen an der Universität Oldenburg

In der Arbeitsgruppe Energiemeteorologie an der Universität Oldenburg wurde die Ausrichtung des AP 3.2 von einer eher passiven Beteiligung an GridFTP-Transfers zwischen dem Datenzentrum am DLR-DFD und der Arbeitsgruppe hin zu einer Evaluierung der Einsatzmöglichkeiten von GridFTP und dCache im Zusammenhang mit umfangreichen Grid-Rechnungen verschoben und erweitert.

Während der Beschäftigung mit GridFTP wurde insbesondere Wert auf einen sicheren Betrieb des eigenen GridFTP-Servers gelegt. Daher wurde ein Sicherheitskonzept erstellt und umgesetzt. Da die Verwendung von GridFTP im Rahmen dieses AP von Anfang an vorgesehen war, wurde bereits frühzeitig ein GridFTP-Server testweise auf einem relativ isolierten System installiert. Einfache Transfertests belegten die Funktionalität, allerdings waren jeweils zusätzliche

Verschiebungen der Daten per `scp` notwendig, weil der Server keinen Zugriff auf die Netzlaufwerke hatte.

Die standardmäßig vorgeschlagene Installation des GridFTP-Servers sieht eine direkte Einbindung ins Dateisystem des Servers vor und regelt alle Zugriffsrechte über die entsprechenden UNIX-Mechanismen. Für dedizierte Grid-Maschinen ist dies eine sinnvolle und ausreichend sichere Einstellung, zumal der Nutzerkreis durch die Abhängigkeit von gültigen Zertifikaten sehr begrenzt ist. Alle diese Nutzer erhalten a priori vergleichbare Rechte, insbesondere der Lesezugriff ist normalerweise sehr umfassend.

Wird allerdings ein an ein internes Network File System (NFS) angebundenes System zum GridFTP-Server aufgerüstet, erhalten alle durch ein (Grid-)Benutzer-Zertifikat zugelassenen Nutzer ggf. Einblick in weitere Verzeichnisse der lokalen Nutzer, in Betriebssystem-spezifische Informationen oder in eingebundene Datenarchive. Denn zur Zeit werden vom GridFTP-Server des GT4 keine „Sandboxen“ unterstützt, wie es bei einigen FTP-Server der Fall ist. Die Nutzergruppen-basierte Verwaltung des Zugriffs über standardmäßige Verzeichnis- und Dateirechte oder auch über Access Control Lists (ACL) würde ein so hohes Maß an Disziplin und Kontrolle erfordern, so dass dieser Ansatz in der Arbeitsgruppe schnell verworfen wurde.

Die Anforderungen an eine sichere und leicht nutzbare Auslegung des GridFTP-Servers wurden zusammengetragen:

- Mitglieder der Gruppe Energiemeteorologie sollen direkt auf Verzeichnisse und Dateien im internen NFS zugreifen können.
- Mitglieder der Gruppe Energiemeteorologie sollen die Funktionalität von GridFTP nutzen können, ohne sich auf einem bestimmten System anmelden zu müssen.
- Nur Mitglieder der virtuellen Organisation WISENT sollen den GridFTP-Server nutzen können.
- Mitglieder von WISENT, die nicht zur Gruppe Energiemeteorologie gehören, sollen nur ein „public“-Verzeichnis sehen und für Dateitransfers nutzen können

Ausgehend von diesen Anforderungen wurde in Zusammenarbeit mit dem OFFIS ein Betriebskonzept entwickelt:

- Der GridFTP-Server wird auf einem operationellen, ins NFS eingebundenen Server installiert.
- Das Installationsverzeichnis liegt unterhalb einer `chroot`-Umgebung, in welche die notwendigen Systemverzeichnisse hineinkopiert oder per `mount` eingebunden werden. Der Rest des Dateisystems ist aus diesem Verzeichnisbaum heraus nicht erreichbar.
- Alle Mitglieder von WISENT, die über ein Benutzer-Zertifikat verfügen und somit GridFTP nutzen können, werden manuell in das `grid-mapfile` eingetragen und zur lokalen Gruppe `wisent` hinzugefügt.

- Die Gruppe `wisent` hat Lese- und Schreibrechte auf einem Verzeichnis „public“ innerhalb der chroot-Umgebung.
- Die Mitglieder der Gruppe Energiemeteorologie werden zusätzlich der Gruppe `condor` hinzugefügt. Die Gruppe wurde im Rahmen der Job-Ausführung mittels Condor bzw. Condor-G eingeführt.
- Die Gruppe `condor` hat Lese- und Schreibrechte auf einem Verzeichnis „restricted“ innerhalb der chroot-Umgebung.
- Alle relevanten Netzlaufwerke werden mit der Option „bind“ unterhalb von „restricted“ gemounted und sind so ausschließlich für die Mitglieder der Gruppe `condor` erreichbar.

Die Umsetzung dieser Eckpunkte gestaltete sich als nicht ganz einfach, insbesondere der Betrieb des GridFTP-Servers innerhalb der chroot-Umgebung ist eigentlich nicht vorgesehen und erfordert einige Vorarbeiten. Die dabei durchgeführten Arbeiten sind in Anhang A aufgeführt.

Die damit geschaffene Möglichkeit, per GridFTP auf das Dateisystem der Arbeitsgruppe Energiemeteorologie an der Universität Oldenburg zuzugreifen, wurde im Rahmen einer größeren Grid-Rechnung ausgenutzt. Dabei werden weitgehend automatisch Daten aus dem Satellitenbild-Archiv der Arbeitsgruppe auf verschiedene Rechencluster kopiert, dort bearbeitet und die Ergebnisse - Karten der solaren Einstrahlung unter verschiedenen Einfallswinkeln - nach einer ersten Zwischenspeicherung auf einem dCache-Server zwecks Visualisierung wieder ins Dateisystem des Benutzers kopiert.

Der Workflow ist in Abbildung 13 dargestellt und zeigt die notwendigen Schritte ausgehend vom Cloudindex (einer Darstellung des Bewölkungsgrades, aus Satellitenbildern extrahiert) bis hin zu einer Monats- oder Jahreskarte, die örtlich aufgelöst anzeigt, unter welchem Winkel die Einstrahlung am besten mit photovoltaischen Anlagen genutzt werden kann.

Die rechteckigen Elemente in Abbildung 13 stellen jeweils ein Datenobjekt dar, die ovalen Elemente stehen für Berechnungen auf Rechenclustern. Das Zwischenprodukt (daily insolation on tilted planes ...) ist speicherplatzintensiv, da für jeden Tag (zunächst eines Jahres, später sogar von 10 Jahren) und jeden Winkel (bisher 49, später deutlich mehr) eine Karte gespeichert werden muß. Diese Daten wurden zentral im dCache-Archiv in Jülich abgespeichert.

Für die Aufsummierung von Tageskarten zu Monats- und Jahreskarten wurden dann gezielt die benötigten Dateien aus dem dCache-Archiv wieder heruntergeladen.

Sämtliche Transfers wurden mit GridFTP durchgeführt. Dabei wurde der File-Staging-Mechanismus von Condor und Globus nur für die verwendeten Programme benutzt, die

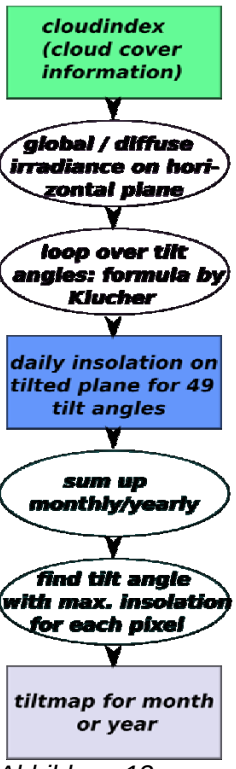


Abbildung 13: Workflow für die Berechnung von Tiltfaktoren

Datentransfers wurden innerhalb eines Perl-Skripts ausgeführt, welches von einer zentralen Schleife mit dem zu bearbeitenden Datum versorgt wird.

Schließlich mussten die berechneten Monats- und Jahreskarten nach der Berechnung wieder auf den Arbeitsplatzrechner verschoben werden. Dafür wurden manuell ausgeführte GridFTP-Befehle verwendet.

In Abbildung 14 sind die Datenwege schematisch dargestellt. Die roten Pfeile entsprechen GridFTP-Transfers, mit den schwarzen Pfeilen sind die Grid-Jobs angedeutet.

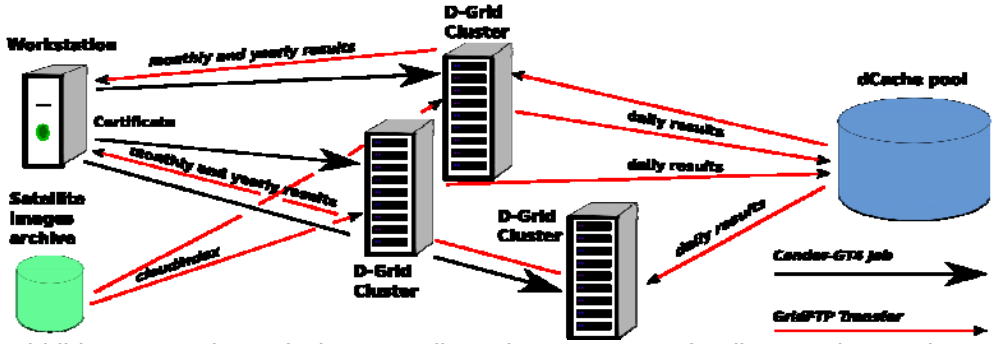


Abbildung 14: Schematische Darstellung der Datenwege für die Berechnung des optimalen Montagewinkels photovoltaischer Anlagen

Auch wenn die endgültige Auswertung der Berechnungen noch nicht abgeschlossen ist, hat sich deutlich abgezeichnet, dass die Durchführung dieser aufwendigen Berechnungen lokal kaum möglich gewesen wäre. Ohne die Möglichkeit der Zwischenspeicherung wäre auch die Bearbeitung im Grid sehr schwierig geworden. Der Einsatz von GridFTP als zuverlässiges und relativ leicht handhabbares Transfer-Werkzeug hat die Umsetzung stark vereinfacht.

Das GridFTP-Konzept hat sich in diesem Kontext sehr bewährt. So war bei gleichzeitiger maximaler Begrenzung des Zugriffs für AG-externe WISENT-Mitglieder möglich, die Ausgangsdaten direkt aus dem Gruppen-eigenen Archiv zu den Rechenclustern zu kopieren, ohne auf Grid-fremde Sonderlösungen wie beispielsweise `secure copy (scp)` ausweichen zu müssen. Damit war insbesondere der enorme Vorteil des `single sign on` erhalten, alle Aktivitäten konnten mit einem einzigen Benutzer-Zertifikat durchgeführt werden.

4.2 dCache

Ein Vorteil der Beteiligung des Projekts WISENT an der D-Grid-Infrastruktur besteht im Zugang zu externen Speicherkapazitäten in den Rechenzentren Jülich, Hannover und Karlsruhe, die über die Datenmanagementsoftware dCache genutzt

werden können. Im Unterschied zum lokalen Festplattenspeicher z. B. im Oldenburger Rechencluster oder im Netzwerk der Forschungsgruppe handelt es sich um beträchtliche Kapazitäten (> 40 TB), die zum Teil über Bandspeicher realisiert werden und sich vorzüglich zur mittelfristigen Datenarchivierung eignen. Diese Speicher werden im D-Grid mit Hilfe der Middleware dCache verfügbar gemacht.

WISENT gehört zu den ersten D-Grid-Projekten, die Anwendungsszenarien für angebotene dCache-Installationen untersucht und effektive Lösungen diesbezüglich gefunden haben. Im Folgenden werden die gesammelten Erfahrungen ausführlich dokumentiert. Zusammenfassend wurde in WISENT ein Softwarewerkzeug und eine Vorgehensweise etabliert, die den wissenschaftlichen Anwendern die Auslagerung ihrer großvolumigen Daten (z. B. mehrere hunderte GB) zu externen D-Grid-Speichersystemen und eine nachträgliche zuverlässige Datenwiederherstellung aus diesen Systemen ermöglicht. So belegte zum Zeitpunkt der Verfassung dieses Berichts WISENT über 3 TB Speicher im Jülicher dCache-Archiv.

dCache bietet mehrere Möglichkeiten (so genannte Doors) an, um auf Daten zuzugreifen. Dazu gehört unter anderem die GridFTP-Door, die prinzipiell mit jedem GridFTP-Client verwendet werden kann. Des Weiteren gibt es die Möglichkeit den Storage Resource Manager (SRM) zu nutzen. Dieser wird über das gleichnamige SRM-Protokoll angesprochen, das im Wesentlichen ein Vermittlungsprotokoll ist. Dabei wird nach einer Dateianfrage (mit dem Befehl srmcp) ein Übertragungsprotokoll ausgehandelt, z. B. GridFTP, und der SRM-Dienst gibt eine URL an, worüber die Datei zu beziehen ist. Letzteres dient unter anderem zum Lastausgleich, wenn eine Datei auf mehreren Servern vorhanden ist.

Trotz des erweiterten Funktionsumfangs von SRM gegenüber GridFTP bietet eine direkte Verwendung der GridFTP-Door aus Anwendersicht einige Vorteile. So kann vertraute GridFTP-Software, die konzeptionell einer herkömmlichen FTP-Software entspricht und sich lediglich in nicht-funktionalen Eigenschaften (positiv) unterscheidet, für Übertragungen von/zu dCache verwendet werden. Im Unterschied zu SRM-Clients kann dieselbe Software auch für Übertragungen zwischen einzelnen Rechenclustern eingesetzt werden, in denen weder dCache noch eine andere SRM-Implementierung vorhanden ist (die meisten zum D-Grid gehörenden Rechencluster fallen in diese Kategorie). Darüber hinaus können Datenübertragungen aus einem Globus Toolkit 4 Berechnungsjob im Rahmen des in WS-GRAM integrierten File Staging zurzeit ausschließlich mit GridFTP durchgeführt werden. Eine Übertragung mit SRM könnte zwar (falls die Client-Software am Ausführungsort verfügbar ist) aus einem bereits ausgeführten Job heraus erfolgen, würde aber zur ineffizienten Nutzung von CPU-Ressourcen und/oder zur Verkomplizierung der in WISENT typischerweise vom Anwender vorgenommenen Job-Programmierung führen. Die Vorteile von SRM gegenüber dem direkten Einsatz von GridFTP wären dagegen die Möglichkeit der Platzreservierung vor einer Datenübertragung und der Regelung von Gültigkeitsdauern der gespeicherten Daten. Diese Vorteile spielen jedoch im folgenden Anwendungsszenario eine untergeordnete Rolle.

4.2.1 Anwendungsszenario und Ausgangssituation

Die grundsätzliche Motivation für die Nutzung von dCache in WISENT ergibt sich aus der relativen Speicherplatzknappheit in lokalen Netzwerken der beteiligten Forschungsinstitute und aus der Notwendigkeit der mittelfristigen Datenarchivierung, um zum Beispiel auf Ergebnisse von vorherigen Berechnungen in der Zukunft zurückgreifen zu können, ohne diese nochmal ausführen zu müssen. Damit ergibt sich die Sicht auf dCache-Installationen als „virtuelle Festplatten“, die auf zuverlässige und performante Weise die üblichen Dateisystemoperationen unterstützen sollen. Beispielsweise soll eine Übertragung eines ausgewählten Verzeichnisbaums aus einem Rechencluster zum dCache-Archiv (und umgekehrt) möglich sein. Darüber hinaus soll es möglich sein, die Inhalte eines vorhandenen Archivs ähnlich wie die eines lokalen Dateisystems anzuschauen und den Platzbedarf für die archivierten Dateien vor einer Übertragung zu ermitteln.

Im Laufe des Arbeitspakets stellte sich heraus, dass die einfachen Kommandozeilenwerkzeuge wie globus-url-copy oder srmcp, deren Merkmale und Verwendungsmöglichkeiten in folgenden Abschnitten noch genauer erläutert werden, für den vorgestellten Zweck nicht ausreichen. Dies ergab die Motivation zur Entwicklung einer neuen Client-Software (siehe Dokument „AP 3.2 – GridFTP-Client“), die im Projektverlauf zum empfohlenen Werkzeug für WISENT-Anwender, die ihre großvolumigen Daten im D-Grid archivieren bzw. wiederherstellen möchten, wurde.

4.2.2 dCache-Performanz

Obwohl wie oben beschrieben aus Anwendersicht die Nutzung von dCache-Servern als „virtuelle Festplatten“ bzw. wie übliche File-Server im LAN wünschenswert ist, die sich von herkömmlichen Dateispeichern so wenig wie möglich unterscheiden sollten, ist in der Praxis dieses Ideal kaum erreichbar. Vielmehr müssen von Anwendern bestimmte aus fachlicher Sicht irrelevante Einschränkungen berücksichtigt werden, die zum einen aus der gegenwärtigen Implementierung von dCache, zum anderen aus inhärenten technischen Merkmalen eines verteilten, auf Bandmedien basierenden Speichersystems resultieren. Insbesondere die Performanz (im Sinne von maximal erreichbaren Datendurchsätzen) wird durch die vom Anwender abhängige Nutzungsweise und nicht nur durch die Netzwerkinfrastruktur stark beeinflusst.

Auf Grund von gesammelten Erfahrungen wurden für WISENT-Anwender folgende Empfehlungen zur Optimierung der Übertragungsperformanz ausformuliert:

1. Die Übertragung von/zu dCache ist schnell (Beispiel: 10-20 MB/s – vergleichbar mit Ergebnissen der allgemeinen GridFTP-Performanztests), wenn die Dateien sich im Pool (d. h. auf der Festplatte des dCache-Systems und nicht auf Band) befinden oder bei Übertragung zum dCache ausreichend Platz im Pool vorhanden ist, um sie aufzunehmen. Beispielsweise ist das Jülicher Archiv so konfiguriert, dass das kritische Datenvolumen 600 GB beträgt, wobei sich diese Datenmenge auf mehrere parallele Nutzer verteilen könnte. Nach der Überschreitung dieser Menge lässt der Datendurchsatz dramatisch nach.

2. Die Übertragung von/zu dCache ist langsam (Beispiel: 1 MB/s), wenn Dateien auf Band geschrieben oder von Band gelesen werden müssen. Man muss in diesem Fall entsprechend viel Geduld aufbringen.
3. Da das Bandsystem pro Datei eine Anlaufzeit von ca. 30 Sekunden hat, ist die Verwendung von Dateigrößen >= 1GB empfohlen – es lohnt sich also, vor der Übertragung zum dCache mehrere Dateien in ein (komprimiertes) Archiv zu packen. Die Speicherung von großen Mengen kleiner Dateien führt dazu, dass ihre spätere Wiederherstellung sehr verlangsamt wird.
4. Dateien sollten nicht sofort nach der Übertragung zum dCache aus dem dortigen System wieder gelöscht werden. Dies verursacht eine „Verschmutzung" des Pools – die Dateien werden dabei nicht wirklich entfernt – und senkt dessen freie Kapazität und damit auch die Performanz von späteren Übertragungen (vgl. Punkt 1). Die Dateien sollten erst dann gelöscht werden, nachdem sie erfolgreich auf Band übertragen worden sind (dies dauert in der Regel mindestens einige Stunden).

4.2.3 Grafischer GridFTP-Client

Speziell für die produktive Nutzung von dCache wurde in OFFIS ein grafischer GridFTP-Client auf der Basis von Eclipse entwickelt: http://bi.offis.de/gridftp

Die Nutzung dieses Clients, besonders für große Übertragungen von/zu dCache, ist empfohlen, da er speziell auf dCache ausgerichtete Fehlertoleranzmerkmale besitzt, die in den Kommandozeilenwerkzeugen fehlen. Es funktioniert nach dem Prinzip „fire and forget", d. h. eine Übertragung wird gestartet und ggf. mehrere Tage lang überwacht, bis sie abgeschlossen ist. Ebenfalls kann eine Übertragung bei Bedarf angehalten und danach (auch mit einem neu erstellten Proxy-Zertifikat) fortgesetzt werden. Bei den primitiven Clients läuft die Übertragung dagegen meist nur bis zum ersten Fehler, wonach sich der Benutzer selbst um die Feststellung und Rettung von halbübertragenen Dateien kümmern muss.

Damit der GridFTP-Client funktioniert, muss der Rechner, auf dem der Client läuft, mit den betroffenen GridFTP-Servern kommunizieren können. Dies bedeutet im D-Grid-Kontext, dass die ausgehende TCP-Kommunikation auf den Ports 20000-25000 und 2811 in der Firewall freigeschaltet sein muss. Im Zweifelsfall wenden sich die Anwender an ihren zuständigen Netzwerkadministrator, um diese Voraussetzung zu ermitteln bzw. zu schaffen.

4.2.4 dCache-Administrationsschnittstelle

Jede dCache-Installation bietet eine Webschnittstelle als Überblick des aktuellen Systemzustands. Diese Schnittstelle stellte sich als nützlich zur Feststellung von manchen Fehlertypen, zur Erklärung von beobachteten Performanzeigenschaften und als zusätzliche Überwachungsmöglichkeit heraus.

Beispielsweise kann man unter der Adresse http://dcache.fz-juelich.de:2288/ genaueres über den aktuellen Zustand von dCache erfahren:

- Unter „Pool Usage" sieht man, wie der Platz im Festplattencache aktuell genutzt wird. Die für WISENT vorgesehenen „Pools" heißen dort „pool1_3" und „pool2_3". Dies entspricht zwei Dateisystemen auf der Festplatte, mit

einer fixen Größe von jeweils 300 GB. Überträgt man eine Datei zum dCache, so wird sie zunächst in einen dieser Pools geschrieben. Möglichst bald nach der erfolgreichen Übertragung wird sie (asynchron und automatisch) von dCache auf Band gespeichert. „Precious Space“ in der Anzeige umfasst diese Dateien, die im Pool liegen, aber noch nicht auf Band gespeichert sind.

- In „Tape Transfer Queue“ sieht man, welche Transfers von Band zum Pool gerade aktiv sind. Die Transfers werden dann aktiviert, wenn man eine Datei herunterladen möchte, die sich nicht mehr auf der Festplatte befindet.

4.2.5 Kommandozeilentools

Die Basiswerkzeuge globus-url-copy und srmcp wurden in der frühen Phase der dCache-Nutzung in WISENT getestet und eignen sich vorrangig für kleine Übertragungen oder zur Feststellung von eventuell vorliegenden Fehlern. Sie werden wegen unkomfortabler Nutzung und unzureichender Zuverlässigkeit nicht für die produktive Nutzung empfohlen. Weitere Details zur Nutzung von globus-url-copy und srmcp befinden sich in Anhang B.

4.2.6 Erkannte Probleme mit dCache (GridFTP-Door)

Während der Entwicklung des GridFTP-Clients in OFFIS wurden einige technische Probleme in der produktiv eingesetzten Version 1.7 von dCache entdeckt, die größtenteils erfolgreich durch eine geschickte Client-Implementierung bewältigt werden konnten, um so ihre Auswirkungen auf Endanwender zu verhindern. Bei den meisten Problemen handelt es sich um Abweichungen von der GridFTP-Spezifikation oder von der Referenzimplementierung aus Globus Toolkit 4. Eine nachhaltige Korrektur dieser Probleme würde Anpassungen im Quellcode von dCache selbst erfordern. Aus diesem Grund wurden die dCache-Entwickler in DESY kontaktiert und die Liste von festgestellten Problemen an sie übermittelt. Alle aufgeführten Punkte wurden als tatsächliche, zum Teil aber bereits bekannte Unzulänglichkeiten von dCache anerkannt.

Die Korrekturen wurden größtenteils als durchführbar klassifiziert, allerdings seien alle Punkte bzgl. des erforderlichen Aufwands nur schlecht abschätzbar:

1. dCache unterstützt im Gegensatz zum GridFTP-Server von Globus Toolkit 4 nicht die Übertragung von mehreren Dateien über dieselbe Datenverbindung. Es werden in der Client-Software bei Versuchen, dieselbe Verbindung für mehrere Übertragungen wiederzuverwenden, Fehlermeldungen wie `->500-globus_xio: System error in writev: Broken pipe<-` berichtet. Praktisch zwingt das den Client, für jede Datei eine neue Verbindung aufzubauen, was nicht nur relativ langsam ist, sondern in einem Testfall mit 20000 Dateien vermutlich zu Folgefehlern führte (z. B. `500 Cannot enter passive mode: java.net.SocketException: Address already in use`).
2. dCache unterstützt nicht den 'RNFS'-Befehl. Damit ist die Umbenennung von einmal übertragenen Dateien und Verzeichnissen nicht möglich.

3. dCache berichtet nicht den Eigentümer von Verzeichnissen und Dateien, sondern nur die Zugriffsrechte, die Größe und den Namen. Dies führt dazu, dass, wenn z. B. „Permission denied“ mit Hinweis auf den Eigentümer berichtet wird, man den Eigentümer (per GridFTP) nicht erfahren kann. Es ist ein Direktzugriff auf das PNFS-Dateisystem von dCache erforderlich, den jedoch normale Anwender nicht haben.
4. Die Zeitstempel der letzten Dateiveränderung werden per GridFTP nicht angezeigt und vermutlich in dCache gar nicht gepflegt. Die aus dCache geholten Dateien bekommen neue Zeitstempel. Wenn man dCache als Datenarchiv verwenden möchte, möchte man aber im Prinzip die Dateien mit unveränderten Attributen einschließlich Zeitstempel herausholen. Ein Umweg besteht darin, die Dateien vor der Archivierung mit dem Programm tar zusammenzufügen.
5. dCache berichtet nur die Zugriffsrechte des Eigentümers, nicht aber die der Unix-Gruppe (die einer virtuellen Organisation im Grid-Kontext entspricht) oder der Welt. Die Gruppenzugriffsrechte sind dennoch wirksam (was positiv zu werten ist). So kann man z. B. `chmod 660` an eine Datei anwenden, aber es wird danach immer noch fälschlicherweise `-rw------` berichtet (Lese- und Schreibzugriff nur für Eigentümer), obwohl jetzt die Gruppenmitglieder die Datei löschen/ändern dürfen.
6. dCache berichtet für Verzeichnisse die Zugriffsrechte `-rw` (ohne das Zugriffsrecht `x` im Gegensatz zum normalen (Grid)FTP-Server). Führt man `chmod 600 verzeichnis` aus (was gemäß der Anzeige nichts ändern sollte), so bleiben die berichteten Zugriffsrechte immer noch `-rw`, aber die Zugriffsrechte für die enthaltenen Dateien wechseln auf `--w`. Diese Dateien sind dann nicht mehr lesbar. Man kann den ursprünglichen Zustand mit `chmod 700 verzeichnis` wiederherstellen, wenn man weiß, was passiert ist. Das Problem besteht also darin, dass dCache die Verzeichnis- und Dateirechte unvollständig und anders als in Unix üblich berichtet.
7. Bei GridFTP-Transfers berichtet dCache den Fortschritt („performance markers“) aus Effizienzgründen nur jede 3 Minuten. Ein normaler GridFTP-Server liefert diese Informationen jedoch im 5-Sekunden-Takt. Die Transfers aus dCache erscheinen deswegen für Anwender oft so, als hätte sich die Übertragung aufgehängt. Das wirkt besonders verunsichernd, wenn man darauf mit interaktiven GridFTP-Clients zugreift. Ab Version 1.8 ist das Benachrichtigungsintervall konfigurierbar. Allerdings ist es laut dCache-Entwicklern nicht empfehlenswert, das Intervall so niedrig wie bei herkömmlichen GridFTP-Servern zu setzen, da es das System überlasten könnte.
8. Rekursives Kopieren mit globus-url-copy (Option -r) von dCache zu einem GridFTP-Server oder nach lokal (file://) ist nicht möglich.
9. dCache unterstützt nicht das „data channel authentication“-Feature, man muss also immer in globus-url-copy die Option `-nodcau` verwenden. RFT (von Globus 4.0.4) funktioniert nicht mit dCache (`500 "dcau N"`

`command not understood`). Das fehlende DCAU-Feature stellt zudem eine potentielle Sicherheitslücke dar.

10. Das Anzeigen von Verzeichnisinhalten in dCache ist um Größenordnungen langsamer als bei einem normalen GridFTP-Server. Gleiches gilt auch für das Löschen von Dateien sowie Löschen/Anlegen von Verzeichnissen. Eine gewisse Beschleunigung wäre laut dCache-Entwicklern durch Tuning-Maßnahmen in zukünftigen Versionen erreichbar, insgesamt ist aber das Problem schwierig, da im Originalentwurf von dCache die Performanz von Metadatenzugriffen nicht ausreichend berücksichtigt wurde. Auch diesbezüglich ist es also empfehlenswert, über die Erhöhung von durchschnittlichen Dateigrößen den Overhead von Metadatenzugriffen zu beschränken. In der HEP-Community, die seit längerer Zeit dCache verwendet, betragen die typischen Dateigrößen ca. 1 GB, neuerdings auch ca. 10 GB. Auf der anderen Seite ist es auch empfehlenswert, dass einzelne Verzeichnisse keine große Anzahl von Dateien enthalten. Die in WISENT bisher üblichen 120 Dateien pro Verzeichnis entsprechen dieser Empfehlung.
11. Es passierte manchmal, dass Dateien in dCache angelegt wurden, die dem root-Benutzer gehörten und deswegen nicht mehr von ihrem tatsächlichen Eigentümer gelöscht werden konnten. Diese Dateien mussten vom dCache-Administrator manuell entfernt werden. Dieses Problem wurde laut Aussage von dCache-Entwicklern hoffentlich in Version 1.8 behoben. Es wurde auf jeden Fall bisher nicht von dCache 1.8 Benutzern berichtet, was aber eventuell auf bereits im Einsatz befindliche Workaround-Skripte in deren Installationen zurückzuführen ist.
12. Es wurde beobachtet, dass nachdem Dateien (zumindest laut PNFS) gelöscht worden sind, der angezeigte Wert von „Precious Space“ in „Disk Usage“ nicht auf 0 zurückging. Dieses Problem hängt vermutlich damit zusammen, dass die als „precious“ markierten Dateien per Default nicht gelöscht werden dürfen (als Schutz gegen einen potentiellen Datenverlust). Man könne aber das Löschen von solchen Dateien durch eine Konfigurationsoption für einen Pool administrativ freischalten.
13. Verbindungsversuche zu dCache scheiterten manchmal in unseren GridFTP-Client. Dieses Problem wurde durch Fehlertoleranz in Form von erneuten Verbindungsversuchen umgangen.
14. Beim Löschen von Dateien wurden von dCache Fehlermeldungen berichtet, obwohl die Dateien erfolgreich gelöscht wurden. Diese Fehlermeldungen wurden in der GridFTP-Client-Implementierung ignoriert, um den Anwender nicht über vermeintliche Fehler zu alarmieren:

```
2008-02-13 09:59:45,840 DEBUG [Worker-0]
org.globus.ftp.vanilla.FTPControlChannel:

Control channel sending: DELE gfs_3_20070405_0600_150.grb

2008-02-13 09:59:45,840 DEBUG [Worker-0]
org.globus.gsi.gssapi.GlobusGSSContextImpl: enter wrap

2008-02-13 09:59:45,841 DEBUG [Worker-0]
org.globus.gsi.gssapi.GlobusGSSContextImpl: exit wrap

2008-02-13 09:59:45,841 DEBUG [Worker-0] org.globus.ftp.vanilla.Reply: read
1st line

2008-02-13 09:59:46,989 DEBUG [Worker-0]
org.globus.gsi.gssapi.GlobusGSSContextImpl: enter unwrap

2008-02-13 09:59:46,990 DEBUG [Worker-0]
org.globus.gsi.gssapi.GlobusGSSContextImpl: exit unwrap

2008-02-13 09:59:46,990 DEBUG [Worker-0] org.globus.ftp.vanilla.Reply: 1st
line: 553 Permission denied, reason:

CacheException(rc=5;msg=Pnfs error : java.lang.IllegalArgumentException:

Failed to remove entry 000100000000000000061A40 : Unknown reason.)

2008-02-13 09:59:46,990 DEBUG [Worker-0]
org.globus.ftp.vanilla.FTPControlChannel:

Control channel received: 553 Permission denied, reason:
CacheException(rc=5;msg=Pnfs error :

java.lang.IllegalArgumentException: Failed to remove entry
000100000000000000061A40 : Unknown reason.)
```

Zusätzlich verwiesen dCache-Entwickler darauf, dass die Version 1.8 von dCache das GETPUT-Feature aus der Version 2 des GridFTP-Protokolls umsetzt, was eine performantere parallele Speicherung von mehreren Dateien ermöglicht. Konkret muss bei der Verwendung von GETPUT die GridFTP-Door an der Datenkommunikation nicht als Vermittler teilnehmen, stattdessen kann sie die Adresse des für eine Datei zuständigen Pool-Rechners an den Client direkt weitergeben. Der für die Unterstützung dieses Feature erforderliche Client-seitige Patch für JGlobus wurde im GridFTP-Client angewendet.

5.Ausblick

Die in diesem Dokument geschilderten Erfahrungen mit Grid-Technologien spiegeln den aktuellen Stand am Ende Projektes WISENT wieder. Weil die Grid-Infrastruktur bis zum Projektende und darüber hinaus genutzt und auch erweitert wird, werden weitere Erfahrungen hinzukommen.

Insgesamt wurden innerhalb von WISENT Grid-Technologien erfolgreich eingesetzt, aber auch deren Grenzen aufgezeigt. Bestimmte Szenarien waren aus sicherheitstechnischen Gründen nur unter hohem Aufwand umsetzbar bzw. zum Teil gar nicht realisierbar. In vielen Fällen mussten „Workarounds“ geschaffen werden, u.a. um den Benutzern eine möglichst einfache Bedienung anbieten zu können, z.B. mit Hilfe des grafischen GridFTP-Clients. Die positiven Erfahrungen

überwiegen aber deutlich den negativen, so dass der Einsatz von Grid-Technologien in der Community Energiemeteorologie nachhaltig gewünscht wird.

Anhang A: Einrichtung eines Grid-FTP-Server in einer chroot-Umgebung

Installiert wurde die Version 4.0.3 des Globus Toolkit auf `srvlx050.uni-oldenburg.de`, einem Debian 4.0-System mit X86_64-Architektur. Eine ausführliche englische Installations- und Konfigurationsanleitung ist auch unter https://bi.offis.de/wisent/tiki-index.php?page=GridFTP-chroot-jail zu finden.

Nach der Erfüllung der Abhängigkeiten (Java, PostgreSQL) und dem Anlegen des Benutzers `globus` sowie der Konfiguration des Datenbank gemäß dem Globus Toolkit-Manual folgte das Erstellen eines neuen Verzeichnisbaums mit einem Wurzelverzeichnis und dem notwendigen Baum an Systemverzeichnissen. Welche Verzeichnisse und Dateien benötigt werden, hängt stark von den Programmen ab, die innerhalb der chroot-Umgebung laufen sollen. Die Abhängigkeit von Systembibliotheken kann mittels `ldd <programm>` ermittelt werden. Detailliertere Informationen liefert der Aufruf `strace <programm>`.

```
mkdir /globus-root
mkdir -p /globus-root/usr/local/globus-4.0.3 /globus-root/etc/grid-security
mkdir -p /globus-root/home/wisent-users /globus-root/var \
/globus-root/usr/bin /globus-root/bin /globus-root/lib \
/globus-root/lib64 /globus-root/root /globus-root/tmp /globus-root/sys \
/globus-root/proc /globus-root/dev /globus-root/usr/lib \
/globus-root/usr/lib64 /globus-root/usr/share /globus-root/home/globus \
/globus-root/tmp /globus-root/public /globus-root/restricted/mntpnt
```

Das Verzeichnis `/globus-root/restricted/mntpnt` wird als mountpoint für das Einhängen eines Netzlaufwerks verwendet. Bei Bedarf müssen daher mehrere Verzeichnisse auf dieser Ebene erstellt werden.

Die meisten Systemdateien können aus dem Wirtssystem kopiert werden:

```
cp /lib/libncurses.so.5 /lib/libdl.so.2 /lib/libc.so.6 /lib/libm.so.6 \
   /lib/libpthread.so.0 /lib/libselinux.so.1 /lib/libsepol.so.1 \
   /lib/libnss_files.so.2 /lib/libnss_nis.so.2 /lib/libnsl.so.1 \
   /lib/libnss_compat.so.2 /lib/libpam.so.0 /lib/libpam_misc.so.0 \
   /lib/libproc-3.2.7.so /globus-root/lib/
cp /lib64/ld-linux-x86-64.so.2 /globus-root/lib64/
cp /usr/lib/libperl.so.5.8 /globus-root/usr/lib/
cp /usr/bin/cut /usr/bin/env /usr/bin/vmstat /globus-root/usr/bin/
cp -r /usr/lib/gconv /usr/lib/jvm /usr/lib/locale /usr/lib/perl5 \
    /usr/lib/perl /globus-root/usr/lib/
cp -r /usr/lib64/jvm /globus-root/usr/lib64/
cp -r /etc/postgresql /etc/pam.d /etc/pam.conf /etc/java-1.5.0-sun \
   /etc/shells /etc/login.defs \
   /globus-root/etc/
cp -r /usr/share/locale /usr/share/perl5 /usr/share/perl \
   /globus-root/usr/share/
```

Einige Systemdateien müssen editiert werden, um die geänderten Pfade innerhalb der `chroot`-Umgebung zu respektieren. Die folgenden Dateien sind beispielhaft und müssen individuell an das vorhandene System angepasst werden. Es folgt ein Beispiel für die Datei `/etc/fstab`.

```
# /etc/fstab: static file system information. - Dummy for chroot jail
#
# <file system> <mount point> <type>  <options>       <dump>  <pass>
proc            /proc         proc    defaults        0       0
/dev/cciss/c0d0p1 /           ext3    defaults,errors=remount-ro 0       1
/dev/cciss/c0d0p2 none        swap    sw              0       0
/dev/hda        /media/cdrom0   udf,iso9660 user,noauto     0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto  0       0
```

Da innerhalb der chroot-Umgebung keine mounts erstellt oder aufgehoben werden, kann /etc/mtab als statische Datei erstellt werden.

```
/dev/cciss/c0d0p1 / ext3 rw,errors=remount-ro 0 0
tmpfs /lib/init/rw tmpfs rw,nosuid,mode=0755 0 0
proc /proc proc rw,noexec,nosuid,nodev 0 0
sysfs /sys sysfs rw,noexec,nosuid,nodev 0 0
procbususb /proc/bus/usb usbfs rw 0 0
udev /dev tmpfs rw,mode=0755 0 0
tmpfs /dev/shm tmpfs rw,nosuid,nodev 0 0
devpts /dev/pts devpts rw,noexec,nosuid,gid=5,mode=620 0 0
```

Auch in /etc/profile müssen Pfade angepasst werden.

```
# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).
PATH="/bin:/usr/bin:/usr/local/bin:/usr/local/globus-4.0.3/bin:\
/usr/local/globus-4.0.3/sbin"
export PS1='# '
export PATH
umask 022
export LD_LIBRARY_PATH=/lib:/lib64:/usr/local/globus-4.0.3/lib:\
/usr/lib:/usr/lib64
export GLOBUS_LOCATION=/usr/local/globus-4.0.3/
. $GLOBUS_LOCATION/etc/globus-user-env.sh
. $GLOBUS_LOCATION/etc/globus-devel-env.sh
export JAVA_HOME=/usr/lib/jvm/java-1.5.0-sun/jre
```

Es folgt die eigentliche Installation des Globus Toolkit.

```
chown globus /globus-root/usr/local/globus-4.0.3 /globus-root/home/globus
su globus
cd /globus-root/usr/local/globus-4.0.3
wget http://www-unix.globus.org/ftppub/gt4/4.0/4.0.3/\
installers/bin/gt4.0.3-x86_64_rhas_4-installer.tar.gz
tar xfz gt4.0.3-x86_64_rhas_4-installer.tar.gz
cd gt4.0.3-x86_64_rhas_4-installer
export GLOBUS_LOCATION=/globus-root/usr/local/globus-4.0.3/
./configure --prefix=$GLOBUS_LOCATION --with-buildopts="--static"
make
make install
```

Als nächstes erfolgt die Initialisierung der Datenbank.

```
su globus


psql -d rftDatabase -f /globus-root/usr/local/globus-4.0.3/\
share/globus_wsrf_rft/rft_schema.sql
```

Die Änderungen in /etc/grid-security geschehen im Wirtssystem, das gesamte Verzeichnis wird in die chroot-Umgebung gemounted. Dadurch wird das Problem divergierender Parallelversionen vermieden. Die Existenz eines gültigen (Grid-)Server-Zertifikates wird vorausgesetzt.

```
mkdir -p /etc/grid-security/certificates /etc/grid-security/grid-mapfiles
cp hostcert.pem hostkey.pem /etc/grid-security/
cp /etc/grid-security/hostcert.pem /etc/grid-security/containercert.pem
cp /etc/grid-security/hostkey.pem /etc/grid-security/containerkey.pem
chown globus /etc/grid-security/containercert.pem \
  /etc/grid-security/containerkey.pem
chmod 400 /etc/grid-security/hostkey.pem

wget
http://eugridpma.org/distribution/current/accredited/tgz/ca_GermanGrid-
1.15.tar.gz

tar xfz ca_GermanGrid-1.15-tar.gz
mv ca_GermanGrid/* /etc/grid-security/certificates/
```

Wie bei einer normalen Globus-Installation wird jedes mit einem Benutzer-Zertifikat versehene Mitglied der VO WISENT einem eigenen UNIX-Nutzer zugewiesen, der zur Gruppe `wisent` (mit der Gruppen-ID 1002 in unserem System) gehört:

```
useradd -g 1002 -K UID_MIN=10000 -c "Johannes Hurka" \
   -d /home/wisent-users/dgws0008 dgws0008
```

Die Option `-K UID_MIN=10000` sorgt dafür, daß die Nutzer-IDs oberhalb von 10000 liegen, so daß es keine Kollisionen mit zentral verwalteten Nutzer-IDs gibt.

Zusätzlich wird auch der Nutzer `globus` in die Gruppe `wisent` aufgenommen.

Die VO-Mitglieder werden z.Z. manuell in `/etc/grid-security/grid-mapfile` eingetragen.

```
...
"/C=DE/O=GridGermany/OU=Carl von Ossietzky Universitaet
Oldenburg/OU=Physik/CN=Johannes Hurka"  dgws0008
...
```

Die Home-Verzeichnisse für die angelegten User werden unter `/home/wisent-users/` angelegt, welches in die `chroot`-Umgebung gemounted wird.

Für den privilegierten Zugriff werden alle berechtigten Benutzer zur Gruppe `globus` hinzugefügt (in der Datei `/etc/group`, die nach jeder Änderung wieder nach `/globus-root/etc` kopiert werden muss).

```
...
globus:x:1001:dgws0008:dgws0009:dgws0010:
wisent:x:1002:globus
...
```

Aufgrund eines Bugs im GridFTP-Server können User auf eine Datei, für die sie keine Benutzerrechte haben, nur dann zugreifen, wenn die Gruppenrechte für die primäre Gruppe des Benutzers ausreichen. Entsprechend müssen die Einträge in `/etc/passwd` so geändert werden, dass die primäre Gruppe der privilegierten Nutzer nicht `wisent` (gid 1002) sondern `globus` (gid 1001) ist.

```
...
dgws0008:x:1125:1001:Johannes Hurka:/home/wisent-users/dgws0008:/bin/sh
...
```

Diese Datei muss ebenfalls nach jedem Update nach `/globus-root/etc/` kopiert werden.

Das Verzeichnis `/globus-root/restricted/` soll nur für Mitglieder der Gruppe `globus` zugänglich sein.

```
chown globus.globus /globus-root/restricted
chmod ug+rwx,o-rwx /globus-root/restricted
```

Stattdessen ist `/globus-root/public` für alle GridFTP-User offen. Da es auch aus dem Wirtssystem beschrieben und gelesen werden kann, können hier beispielsweise Daten temporär abgelegt werden, die von nicht-privilegierten GridFTP-Nutzern heruntergeladen werden dürfen.

```
chown globus.wisent /globus-root/public
chmod ugo+rwx /globus-root/public
```

Der komplette Prozess zum Starten und Stoppen des GridFTP-Servers wird von einem angepassten init-Skript `/etc/init.d/gridftp` verwaltet, welches sowohl die Vorbereitung der `chroot`-Umgebung übernimmt (Kopieren von Dateien, Mounten von Verzeichnissen), als auch den GridFTP-Server und den Globus-Container startet. Es kann von `root` manuell mit den Argumenten `start`, `stop` und `restart` aufgerufen werden, aber auch für den Start beim Boot-Vorgang in die entsprechenden Runlevel verlinkt werden. Im Wesentlichen besteht das Skript aus den Funktionen `do_start()` und `do_stop()`, die restliche Funktionalität entspricht einem Standard-Init-Skript.

```
# Function that starts the service
do_start()
{
    # Return 1 if service is already running
    if pgrep globus-gridftp > /dev/null; then
        return 1
    fi
    # Else start with mounting needed dirs:
    mount | grep /globus-root/etc/grid-security > /dev/null || mount -o
bind /etc/grid-security /globus-root/etc/grid-security
    mount | grep /globus-root/etc/default > /dev/null || mount -o bind
/etc/default /globus-root/etc/default
    mount | grep /globus-root/home/wisent-users > /dev/null || mount -o
bind /home/wisent-users /globus-root/home/wisent-users
    mount | grep /globus-root/dev > /dev/null || mount -o bind /dev
/globus-root/dev
    mount | grep /globus-root/sys > /dev/null || mount -o bind /sys
/globus-root/sys
    mount | grep /globus-root/etc/proc > /dev/null || mount -o bind /proc
/globus-root/proc
    mount | grep /globus-root/var > /dev/null || mount -o bind /var
/globus-root/var
    mount | grep /globus-root/tmp > /dev/null || mount -o bind /tmp
/globus-root/tmp
    mount | grep /globus-root/restricted/mntpnt > /dev/null || mount -o
bind /restricted-data /globus-root/restricted/mntpnt
    # And more data mounts if needed
    # Copy needed files:
    cp /etc/hosts /etc/resolv.conf /globus-root/etc
    cp /etc/localtime /etc/passwd /etc/group /globus-root/etc
    cp /etc/shadow /etc/nsswitch.conf /globus-root/etc/
    # Start the server from the chrooted system:
    # Return 2 if service could not be started
    cd /globus-root
    chroot /globus-root /usr/local/bin/start_gridftp_server.sh || return 2
    # Wait a moment - don't know whether it's really necessary...
```

```
    sleep 1
}


# Function that stops the service
do_stop()
{
    if pgrep globus-gridftp > /dev/null; then
        # Kill the globus container
        # (using pkill to search for first part of command line):
        if pgrep -u globus -f "java -Dlog4j.configuration=container-log4j"
> /dev/null; then
            pkill -u globus -f "java -Dlog4j.configuration=container-
log4j"
        fi
        # Stop the server:
        kill -9 `pidof globus-gridftp-server` || return 2
        # Wait a moment for resources to recover before unmount
        sleep 1
        # Clean up mounted dirs:
        umount /globus-root/etc/grid-security > /dev/null
        umount /globus-root/etc/default > /dev/null
        umount /globus-root/home/wisent-users > /dev/null
        umount /globus-root/sys > /dev/null
        umount /globus-root/proc > /dev/null
        umount /globus-root/var > /dev/null
        umount /globus-root/tmp > /dev/null
        umount /globus-root/dev > /dev/null
        umount /globus-root/restricted/mntpnt > /dev/null
        return 0
    else
    # Return 1 if daemon was already stopped
        return 1
    fi
}
```

Wenn `/etc/init.d/gridftp start` aufgerufen wird, werden zunächst die Verzeichnisse und Dateien vorbereitet. Dann wechselt das Skript in das chroot-Verzeichnis `/globus-root` und ruft `chroot` in Verbindung mit dem Skript `/usr/local/bin/start_gridftp_server.sh` auf (im normalen System unter `/globus-root/usr/local/bin/start_gridftp_server.sh` zu finden), welches dann den GridFTP server sowie den Globus-Container startet.

```
#!/bin/bash
# Starts the GridFTP server inside the chroot
. /etc/profile
/usr/local/globus-4.0.3/sbin/globus-gridftp-server -S -p 2811
su globus "-c /usr/local/globus-4.0.3/bin/globus-start-container >
/dev/null&"
```

Mit dieser Konfiguration konnten alle von der Arbeitsgruppe gewünschten Eigenschaften umgesetzt werden.

Anhang B: Einsatz von globus-url-copy und srmcp

zum Zugriff auf dCache

Kopieren einzelner Dateien von lokal nach dCache oder umgekehrt

Dazu kann sowohl globus-url-copy als auch srmcp verwendet werden, z.B.:

```
globus-url-copy -vb gsiftp://dcache.fz-juelich.de/pnfs/fz-
juelich.de/data/wisent/file.img file:///tmp/file.img

globus-url-copy -vb file:///tmp/file.img gsiftp://dcache.fz-
juelich.de/pnfs/fz-juelich.de/data/wisent/file.img

srmcp -debug=false srm://dcache.fz-juelich.de:8443/pnfs/fz-
juelich.de/data/wisent/file.img file:////tmp/file.img

srmcp -debug=false file:////tmp/file.img srm://dcache.fz-
juelich.de:8443/pnfs/fz-juelich.de/data/wisent/file.img
```

Mit der Option `-vb` bei globus-url-copy wird zusätzlich die Datentransferrate angezeigt. Zu beachten ist die Anzahl der Schrägstriche (*/*) nach dem Protokoll `file:`. Bei globus-url-copy sind es 3 (`file:///tmp/file.img`) und bei srmcp 4 (`file:////tmp/file.img`). Der Befehl srmcp ist auch mit der Option `-debug=false` relativ geschwätzig. Folgender Output beim Kopieren einer Datei ist normal:

```
srmcp -debug=false srm://dcache.fz-juelich.de:8443/pnfs/fz-
juelich.de/data/wisent/file.img file:////tmp/file.img

set::trying to set unknown name "storagetype" to "permanent"

set::trying to set unknown name "server_mode" to "passive"

set::trying to set unknown name "long_ls_format" to "false"

set::trying to set unknown name "recursion_depth" to "0"

set::trying to set unknown name "srm_protocol_version" to "1"

set::trying to set unknown name "request_lifetime" to "86400"

user credentials are: /C=DE/O=GridGermany/OU=OFFIS e.V./OU=Grid-RA/CN=Guido
Scherp

SRMClientV1 : connecting to srm at httpg://dcache.fz-
juelich.de:8443/srm/managerv1

org.globus.ftp.exception.ServerException: Server refused performing the
request. ...

GridftpClient:  transfer exception

org.globus.ftp.exception.ServerException: Server refused performing the
request. ...

copy failed with the error

org.globus.ftp.exception.ServerException: Server refused performing the
request. ...

 try again

sleeping for 10000 before retrying
```

Warum diese Fehlermeldung einmal auftritt, konnte nicht geklärt werden. Danach wird der Transfer aber erfolgreich gestartet, was leider nicht berichtet wird. Mit der

Option `-debug=true` erkennt man den Start eines Transfers, hat aber etwas mehr Output.

Kopieren ganzer Verzeichnisse von lokal nach dCache

Ganze Verzeichnisse können mit globus-url-copy kopiert werden, aber nur von lokal nach dCache und nicht umgekehrt. Der Befehl srmcp unterstützt derzeit nur das Kopieren von einzeln genannten Dateien. Zum Kopieren eines ganzen Verzeichnisses bietet globus-url-copy die Option `-r` an, z.B.

```
globus-url-copy -r -vb file:///tmp/verzeichnis/ gsiftp://dcache.fz-
juelich.de/pnfs/fz-juelich.de/data/wisent/temp/
```

Die umgekehrte Richtung erzeugt folgende Fehlermeldung:

```
globus-url-copy -r -vb gsiftp://dcache.fz-juelich.de/pnfs/fz-
juelich.de/data/wisent/temp/ file:////temp/verzeichnis/

error: No files matched the source url.
```

Third-Party-Transfers

Third-Party-Transfers, d. h. Übertragungen von einem GridFTP-Server zum anderen (nicht von einem GridFTP-Server zum Client), sind mit globus-url-copy oder mit srmcp möglich. Im ersten Fall muss die Option `-nodcau` verwendet werden. dCache verhält sich dabei wie ein GridFTP-Client und spricht einen externen GridFTP-Server an. Auch in RFT und WS-GRAM (rftOptions), die bei der Ausführung von Globus Toolkit 4 Jobs zwecks File Staging verwendet werden, kann eine äquivalente Option zu `-nodcau` angegeben werden. RFT funktioniert allerdings auf Grund eines Bugs nicht, der von WISENT an die Globus-Entwickler berichtet wurde: http://bugzilla.mcs.anl.gov/globus/show_bug.cgi?id=5829

Ein File-Staging mit dCache (basierend auf WS-GRAM) wäre also vermutlich auch unmöglich, was aber noch nicht getestet wurde.

Ein Third-Party-Transfer mit globus-url-copy kann wie folgt aufgerufen werden:

```
globus-url-copy -vb -nodcau gsiftp://srvgrid01.offis.uni-
oldenburg.de/tmp/file.img gsiftp://dcache.fz-juelich.de/pnfs/fz-
juelich.de/data/wisent/file.img

globus-url-copy -vb -nodcau gsiftp://dcache.fz-juelich.de/pnfs/fz-
juelich.de/data/wisent/file.img gsiftp://srvgrid01.offis.uni-
oldenburg.de/tmp/file.img
```

Da dCache nur alle 3 Minuten den Forschritt eines Transfers berichtet, wird bei der Option „-vb“ entsprechend nur alle 3 Minuten eine Aktualisierung angezeigt.

Ein ThirdPartyTransfer mit srmcp kann wie folgt aufgerufen werden:

```
srmcp --debug=false srm://dcache.fz-juelich.de:8443//pnfs/fz-
juelich.de/data/wisent/file.img gsiftp://srvgrid01.offis.uni-
oldenburg.de//tmp/file.img

srmcp -debug=false gsiftp://eridanus.caf.dlr.de//tmp/file.img
srm://dcache.fz-juelich.de:8443//pnfs/fz-juelich.de/data/wisent/file4.img
```

Zu beachten ist, dass bei dem GridFTP-Ziel nach dem Hostnamen zwei Schrägstriche (/) angegeben werden müssen (`gsiftp://srvgrid01.offis.uni-oldenburg.de//tmp/file.img`). Das Kopieren ganzer Verzeichnisse per ThirdParyTransfer mit srmcp ist nicht möglich.

Wenn man als Zieladresse ein Verzeichnis angibt, kann man auch mehrere Quelldateien mit einem srmcp-Aufruf transferieren:

```
srmcp --debug=false srm://dcache.fz-juelich.de:8443//pnfs/fz-
juelich.de/data/wisent/file1.img srm://dcache.fz-juelich.de:8443//pnfs/fz-
juelich.de/data/wisent/file2.img gsiftp://srvgrid01.offis.uni-
oldenburg.de//tmp/
```

srmcp ist auf jeden Fall auf Rechenknoten in Oldenburg, Jülich und Hannover verfügbar und wurde dort getestet. Vor der ersten Verwendung von srmcp musste eventuell noch `mkdir ~/.srmconfig` ausgeführt werden.