

# SLA4D<sup>2</sup>GRID

## Architektur des SLA-Frameworks und Integration

### D 2.1

Arbeitspaket AP 2:           Architektur und Integration

Veröffentlichungsdatum: 17/11/2009

Verantwortlicher Editor: Axel Tenschert

Version:                       1.1

Status:                         Review

GEFÖRDERT VOM



Bundesministerium  
für Bildung  
und Forschung

## Autoren

Axel Tenschert, Höchstleistungsrechenzentrum Stuttgart

Roland Kübert, Höchstleistungsrechenzentrum Stuttgart

Dominic Battré, TU Berlin

Oliver Wäldrich, FhG SCAI

Alexander Willner, UBO

## Internes Review

Philipp Wieder, TUDO

## Versionshistorie

<b>Version</b>	<b>Datum</b>	<b>Kommentar</b>	<b>Autoren</b>
0.1	03.07.2009	Inhaltsverzeichnis und erster Vorschlag einer Architektur	Axel Tenschert
0.2	06.07.2009	Referenzen, UML-Kommentar, neues Dateiformat	Alexander Willner
0.3	30.07.2009	Inhaltliche Erweiterungen / erste Komponentenbeschreibungen	Axel Tenschert
0.4	20.08.2009	Inhaltliche Erweiterungen	Dominic Battré
0.5	09.09.2009	Anpassung der Architektur	Axel Tenschert
0.6	29.09.2009	Erweiterung des Dokumentenumfangs und inhaltliche Überarbeitung	Axel Tenschert
0.7	10.10.2009	Inhaltliche Erweiterungen	Dominic Battré, Oliver Wäldrich
0.8	27.10.2009	Inhaltliche Überarbeitung	Axel Tenschert, Roland Kübert
0.9	17.11.2009	Bereich Netze	Alexander Willner
1.0	24.11.2009	Inhaltliche Überarbeitung der Architektur	Axel Tenschert

1.1	01.12.2009	Einarbeitung Review; Hinzufügen der Roadmap	Philipp Wieder
-----	------------	---	----------------

## Kurzzusammenfassung

Im Rahmen des SLA4D-Grid-Projektes wird eine Service Level Agreement (SLA)-Schicht entwickelt, die es Nutzern, Communities, sowie Ressourcenanbietern erlaubt, SLAs auszuhandeln und ihre Einhaltung zur Laufzeit zu überprüfen. Somit soll es möglich gemacht werden, Dienstgütern zu garantieren, deren Erfüllung bezahlen zu lassen und deren Verletzung mit Strafen zu belegen.

Dieses Dokument beschreibt den ersten Entwurf der Architektur des SLA-Frameworks, das im Rahmen des SLA4D-Grid-Projektes entwickelt werden soll. Die Dienste, die gemäß dieser Architektur implementiert werden, werden in das existierende D-Grid integriert, daher ist es essentiell, deren Einordnung in das D-Grid zu betrachten und Integrationsaspekte zu diskutieren, was einleitend in dem vorliegenden Dokument geschieht. Darauffolgend wird der generische D-Grid Anwendungsfall beschrieben.

Den Hauptteil des Projektberichtes macht die Architekturbeschreibung, basierend auf einem Komponentendiagramm, aus. Sämtliche zu entwickelnden Dienste und Komponenten werden hierbei im Detail beschrieben.

Des Weiteren gibt das Dokument eine erste grobe Einschätzung der zu erledigenden Arbeiten und beschreibt deren zeitlichen Ablauf.

## Inhaltsverzeichnis

1	Einleitung.....	6
2	Einordnung in das D-Grid.....	7
3	Integrationsaspekte.....	8
4	Reservierung von Ressourcen – der generische Anwendungsfall.....	9
5	Schnittstellen zwischen den Arbeitspaketen.....	10
6	Architektur des SLA Frameworks.....	12
6.1	Anforderungen an die Architektur.....	12
6.2	Umsetzung der Architektur.....	13
6.2.1	Komponente der SLA Preselection.....	13
6.2.2	Komponenten der Community.....	15
6.2.3	Komponente für SLA-Verhandlung und -Überwachung.....	17
6.2.4	Konkrete Implementierungen.....	20
7	Arbeitsplan bis Projektmonat 12.....	21
7.1	Aktueller Stand der Entwicklung.....	21
7.2	Ziele für Projektmonat 12 (Meilenstein MEI-03).....	22
8	Anmerkungen.....	22
	Referenzen.....	23

# 1 Einleitung

Ziel des SLA4D-Grid-Projektes ist die Entwicklung einer Service Level Agreement Management Schicht (SLA-Schicht) im D-Grid. Mittels dieser Schicht sowie weiteren Entwicklungen für die momentane D-Grid Infrastruktur soll bestehenden und zukünftigen D-Grid-Communities sowie Diensteanbietern die Möglichkeit gegeben werden, ökonomische Aspekte in die im D-Grid angebotenen Dienste einbringen und entsprechende Geschäftsmodelle für die Plattform D-Grid zu entwickeln. Abbildung 1 zeigt die geplante D-Grid SLA Infrastruktur. Die blau hinterlegten Komponenten werden vom SLA4D-Grid-Projekt entwickelt.

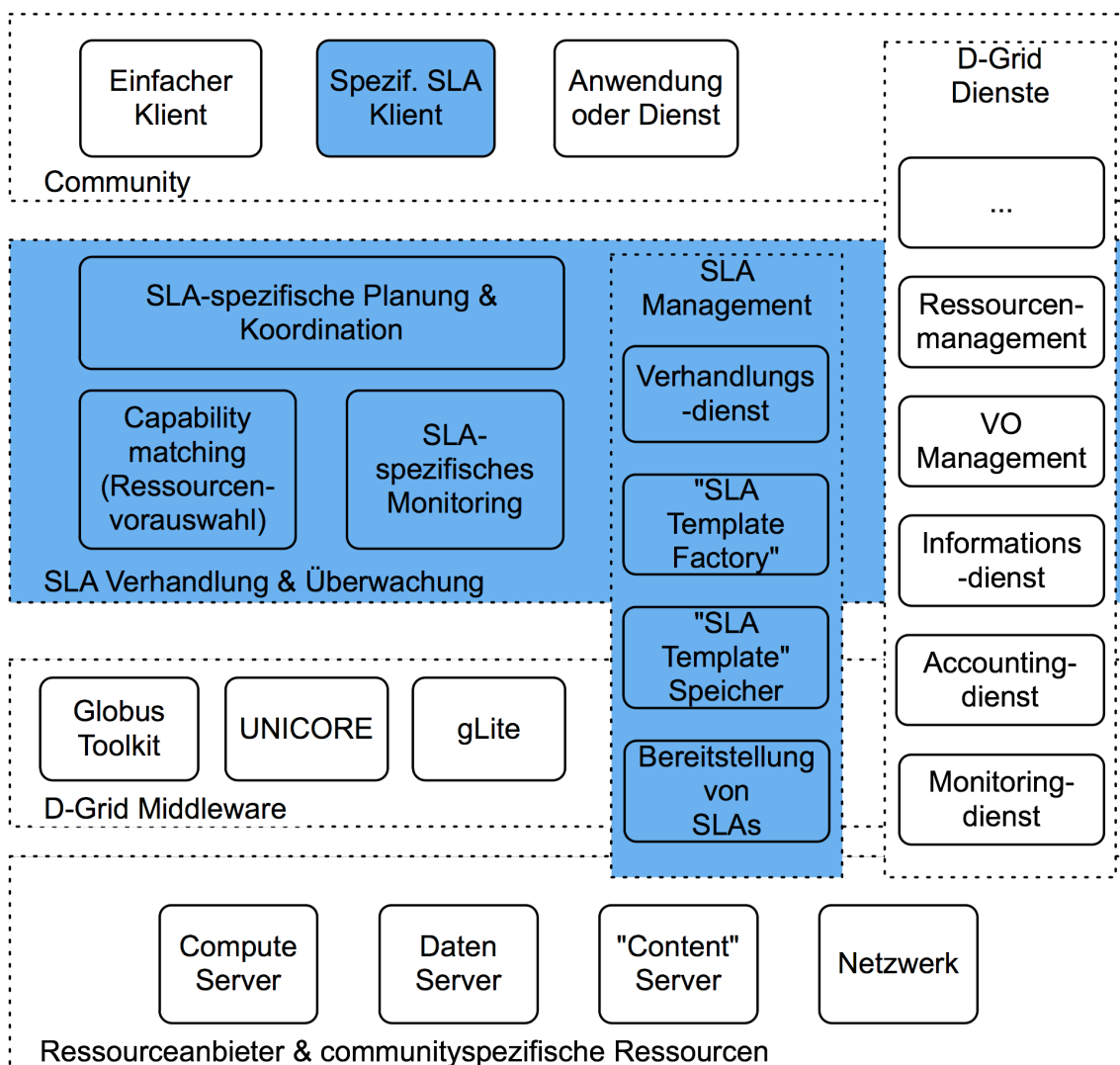


Abbildung 1: D-Grid SLA Management Schicht

Die Dienste und die klientenseitige Anbindung werden über die Projektlaufzeit entwickelt und integriert, verschiedene Iterationen der Architektur und der Dienste werden zum

Projektabschluss in einer für die Communities und Ressourcenanbieter des D-Grid nutzbaren Infrastruktur resultieren. Detaillierte Angaben zu den Entwicklungen des ersten Prototyps bis Projektmonat 12 (Mai 2010) finden sich in Abschnitt 7.

## 2 Einordnung in das D-Grid

Das SLA4D-Grid-Projekt findet im Rahmen der D-Grid-Initiative statt. Hierbei wird eine SLA-Schicht angestrebt, die den unterschiedlichen Anforderungen im Rahmen von D-Grid gerecht wird.

Zu diesem Zweck wird im Rahmen des Projektes ein generisches D-Grid SLA definiert und um community-spezifische Anforderungen erweitert (siehe auch den entsprechenden SLA4D-Grid Projektbericht [1]). Das generische SLA bildet das grundlegende Nutzungsszenario im D-Grid, die Verwendung von Rechnerressourcen, ab und erlaubt deren Nutzung unter wirtschaftlichen Rahmenbedingungen. In einer Vielzahl von Projekten der D-Grid Community spielt dieses Szenario eine Rolle, so dass es als grundlegender, erweiterbarer Anwendungsfall gewählt wurde. Gleichzeitig und parallel zu den Entwicklungen der SLA-Schicht diskutiert SLA4D-Grid mit anderen Projekten die aktuellen Entwicklungen und deren potentielle Anwendung im jeweils anderen Projekt. So ist beispielsweise die Reservierung von Rechnerressourcen ebenfalls ein Anwendungsfall im DGSi Projekt. Hier werden Anforderungen und Angebot verglichen und potentielle neue Features bzgl. ihrer Einbringung in die SLA-Schicht evaluiert.

Community-spezifische Anforderungen bezüglich der SLA-Schicht ergeben sich zum Zeitpunkt der Erstellung dieses Berichtes aus folgenden Anwendungsszenarien:

- Den beiden kommerziellen Anwendungsszenarien des SLA4D-Grid Vorhabens, des der con terra GmbH sowie dem von Sun Microsystems,
- dem Anwendungsszenario „Delegation von Ressourcen“ des DGSi Projektes,
- sowie Community-Anwendungen, die mittels Fragebögen und durch Workshops evaluiert werden.

An dieser Stelle ist festzuhalten, dass im Rahmen des Projektes SLA4D-Grid nicht alle Anforderungen der diversen Anwendungsszenarien umgesetzt werden können, da dazu weder der Zeitrahmen noch die Ressourcen ausreichen. Allerdings ergibt die Auswertung der Szenarien ein umfassendes Bild der Community-Anforderungen und die Möglichkeit, generische Grundlagen zu deren Umsetzung im Rahmen des D-Grid zu schaffen.

Das übergeordnete Ziel des Projektes SLA4D-Grid ist es hierbei, einen Verbund von Diensten zu entwerfen und zu realisieren, der eine Service Level Agreement Management Schicht für das D-Grid bereitstellt. Diese Schicht ist, wie in Abbildung 2 dargestellt, zwischen der D-Grid-Middleware (gLite<sup>1</sup>, Globus<sup>2</sup> und UNICORE<sup>3</sup>) und der Benutzerebene des D-Grid anzusiedeln und wird in die bestehende Infrastruktur integriert (unter anderem durch Anbindung an ausgewählte Kern-D-Grid-Dienste). Communities können die SLA-Schicht entweder über spezielle Klienten oder aber eine Schnittstelle nutzen. Da die SLA-Schicht einen Mehrwertdienst darstellt, beeinflusst sie die gegenwärtige Nutzung des D-Grid nicht, sondern greift bei Communities ohne Bedarf nach SLAs nicht in den Betrieb ein

<sup>1</sup> <http://glite.web.cern.ch/glite/common/introduction.asp>

<sup>2</sup> <http://globus.org/>

<sup>3</sup> <http://www.unicore.eu/>

(wie in Abbildung 2 durch *Community A* dargestellt). Auf diese Weise ist eine nahtlose Einführung der SLA-Schicht in den Regelbetrieb möglich. Zudem bietet die SLA-Schicht sowohl den Benutzern, ganzen Communities, als auch den Anbietern von D-Grid-Ressourcen eine Gewährleistung der Ressourcennutzung unter wirtschaftlichen Bedingungen. Zu diesem Zweck werden Dienstgüternachfragen und die korrespondierenden Angebote durch Service Level Agreements verbindlich zusammengeführt.

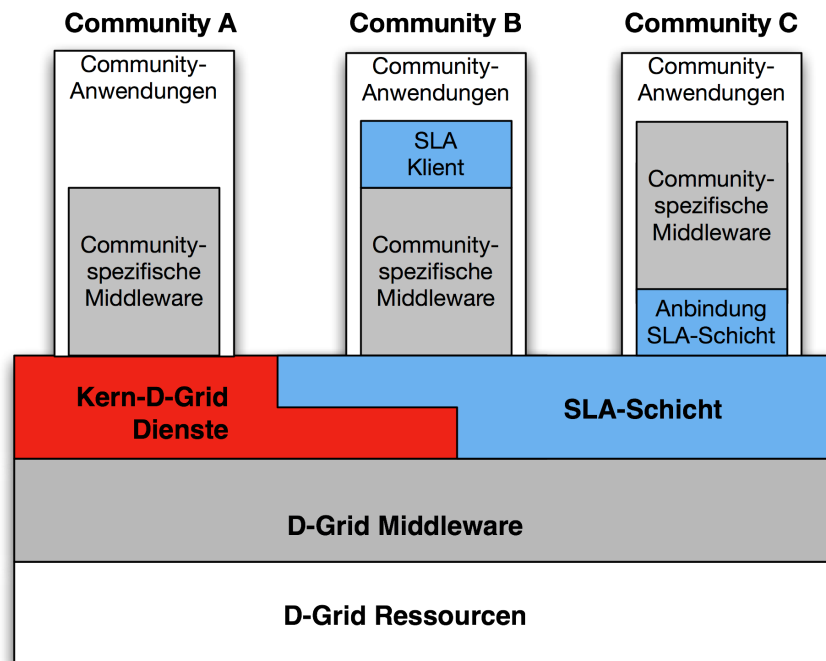


Abbildung 2: Generelle Einordnung der SLA-Schicht bezüglich des D-Grid

Mittels der SLA-Schicht und unter Zuhilfenahme weiterer D-Grid-Dienste wie beispielsweise Monitoring oder Accounting können SLAs automatisch erstellt, verhandelt und ihre Einhaltung überwacht werden, so dass das D-Grid von akademischen und industriellen Nutzern gemäß ihrer jeweiligen Geschäftsmodelle in einer ökonomisch effizienten Weise genutzt werden kann.

Die in Abbildung 2 dargestellte Anbindung der Communities an die Kern-D-Grid Dienste, bzw. an die SLA-Schicht gibt hierbei nur einen Überblick. Die Anbindung an die D-Grid-Infrastruktur wird im Detail in AP1 behandelt.

### 3 Integrationsaspekte

Während des gesamten Projektverlaufs wird ein integrativer Ansatz verfolgt. Hierdurch wird eine enge Kooperation der einzelnen Partner und zwischen den Arbeitspaketen sichergestellt. Dabei sind sämtliche Partner in die Entwicklung der Architektur für die SLA Schicht mit einbezogen.

Die einzelnen Komponenten des SLA Frameworks werden mit Hilfe einer gemeinsamen Versionsverwaltung entwickelt und täglichen Funktions- und Integrationstests unterzogen, um frühzeitig Probleme zu erkennen und zu beheben. Das Maven-Build-Management-

Tool<sup>4</sup> ist für automatisierte Tests sowie die interne und externe Versionierung von Komponenten zuständig. Somit können bereits während der Projektlaufzeit Revisionsstände veröffentlicht und ausgetauscht werden.

Um die Integration der SLA Schicht mit bestehenden und noch zu entwickelnden Komponenten des D-Grid sicherzustellen, werden die Anforderungen an die SLA Schicht aus anderen Community Projekten im Rahmen der Arbeitspakete 1 und 6 erhoben. Am 03.09.2009 hat bereits ein Workshop zur Anforderungserhebung stattgefunden. Hier haben Mitglieder der Projekte ValueGrids, Services@MediGRID, WissGrid, PT-Grid, BIS-Grid, OptiNum-Grid und GDI-Grid teilgenommen. Die Ergebnisse dieses Workshops sind im Projektbericht D6.1 [1] festgehalten.

## 4 Reservierung von Ressourcen – der generische Anwendungsfall

Entsprechend der Anforderungserhebung aus dem Szenario des Geodateninformationssystems der con terra GmbH (siehe Projektbericht D3.1 [1]) ist die Reservierung von Ressourcen sowie die anschließende Ausführung eines Programms auf diesen Ressourcen die grundlegende Funktion, die der erste Prototyp der SLA-Schicht bereitstellen wird. Die Reservierung von Ressourcen ist somit der generische Anwendungsfall (der mit dem generischen D-Grid SLA korrespondiert).

Um die anzufordernden Ressourcen mittels eines SLAs zu spezifizieren, wurde zusammen mit Arbeitspaket 3 ein auf dem WS-Agreement Standard basierendes generisches D-Grid SLA definiert. Dieses erweitert WS-Agreement um die so genannten „Advanced Reservation Term Language“ (siehe hierzu ebenfalls D3.1), die es erlaubt, grundlegende Parameter zur Reservierung zu definieren. Mittels der Job Submission and Description Language (JSDL) ist es zudem möglich, zu spezifizieren, welches Programm (welcher Job) innerhalb der Reservierung zur Ausführung gebracht werden soll.

Die Aufgabe der SLA-Schicht soll es nun sein, die im SLA beschriebene Reservierung im Ressourcen Management System zu realisieren und den beschriebenen Job auszuführen. Dazu ist bei jedem der drei im D-Grid verwendeten Middleware ein anderer Ansatz notwendig. Die Middleware UNICORE besitzt bereits nativ die Möglichkeit, Ressourcenreservierungen durchzuführen. Hier ist primär die Umsetzung der Reservierung zu realisieren. Für das Globus Toolkit existiert diese Möglichkeit nicht. Innerhalb Arbeitspaket 2 wurde daher entschieden, basierend auf dem Scheduler HARC<sup>5</sup> [3] einen Dienst zur Ressourcenreservierung zu entwickeln. In diesem Fall wird die SLA-Schicht mit HARC kommunizieren, welches wiederum die Reservierung im RMS<sup>6</sup> umsetzt. Bezüglich der dritten verwendeten Middleware gLite, ist zum jetzigen Zeitpunkt die exakte Umsetzung der Reservierung noch nicht entschieden, wahrscheinlich ist aber die Erweiterung des CREAM CE um einen Reservierungsdienst.

Die Architektur und die Umsetzung der einzelnen Reservierungsdienste, ihre Anbindung an die SLA-Schicht und an die D-Grid Dienste wird detailliert im Projektbericht D2.2 „Prototyp I der SLA-Schicht“ beschrieben werden. Dieser Projektbericht korrespondiert mit

<sup>4</sup> <http://maven.apache.org/>

<sup>5</sup> HARC = Highly-Available Resource Co-allocator

<sup>6</sup> RMS = Resource Management System

dem Meilenstein MEI-03, der laut Vorhabensbeschreibung in Projektmonat 12 (Mai 2010) erreicht werden wird.

Unabhängig von der eingesetzten Middleware soll es weiterhin möglich sein, zusätzlich benötigte Ressourcen zu reservieren, um Dienstgütern auf einer ganzheitlichen Basis zusichern und verhandeln zu können. Dies beinhaltet exemplarisch die Verwaltung von Netzwerkkomponenten zwischen beteiligten Rechen- und Speicherressourcen. Das Netzwerkreservierungssystem ARGON<sup>7</sup> [4] soll hier beispielhaft eingesetzt werden, um die Aspekte der Aushandlung, Reservierung und Beobachtung prototypisch umzusetzen.

## 5 Schnittstellen zwischen den Arbeitspaketen

Arbeitspaket 2 definiert die Architektur des SLA4D-Grid-Projektes und koordiniert die integrativen Aspekte des SLA4D-Grid Vorhabens. Hierzu werden innerhalb von Paket 2 die Schnittstellen zu den weiteren Arbeitspaketen, insbesondere zu AP3, AP4 und AP5, berücksichtigt (siehe dazu auch Abbildung 3).

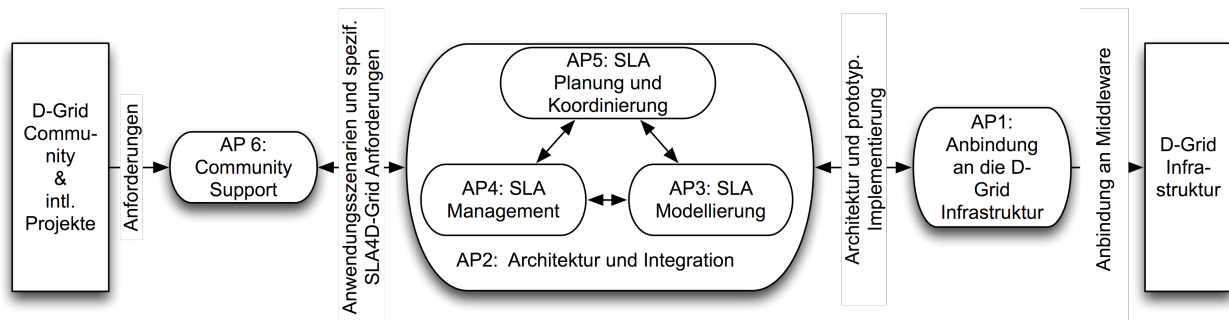


Abbildung 3 Arbeitspakete und ihre Schnittstellen

Arbeitspaket 3 hat im Rahmen eines Workshops am 03.09.2009 und darüber hinaus im Projektbericht „Initial Version of D-Grid SLA“ [1] Szenarien und Anforderungen an die umzusetzenden Dienstgütevereinbarungen definiert. Dies sind im Besonderen die Unterstützung von Ressourcenreservierungen mittels des generischen D-Grid SLAs sowie die Unterstützung der JSDL zur Beschreibung von Ressourcenanforderungen und der auszuführenden Anwendung. Die konkrete Strukturierung der SLAs kann später sehr einfach angepasst werden. Domänenspezifische Erweiterungen des generischen D-Grid SLAs werden hierbei über entsprechende Erweiterungen bereitgestellt. Dabei ist zu erwarten, dass innerhalb der D-Grid Community bei wachsender Anwendung von SLAs verschiedenartige Erweiterungen des generischen SLAs definiert und bestenfalls auch den Communities zur Verfügung gestellt werden.

Arbeitspaket 4 benötigt den Zugriff auf SLA Templates von Ressourcen- und Dienst Anbietern. Die WS-Agreement-Spezifikation [5] schreibt vor, dass Templates als Resource Properties über die AgreementFactory exponiert werden. Damit ist sichergestellt, dass der Zugriff über standardisierte Interfaces erfolgen kann. Neben dem Zugriff gemäß der WSRF-Spezifikation soll es zudem möglich sein, sich über WS-

<sup>7</sup> ARGON = Allocation and Reservation of Grid-enabled Optical Networks

Notification über Änderungen der verfügbaren SLA Templates informieren zu lassen. Auch hier regelt ein anerkannter Standard den Zugriff.

Arbeitspaket 5 konzentriert sich in den ersten 12 Monaten des Projektes auf das Design und die Implementierung eines SLA Verhandlungsdienstes. Dieser Dienst erlaubt es, mit Nutzern Dienstgütegarantien in Form von SLAs zu verhandeln. Zu diesem Zweck wird ein Verhandlungsprotokoll implementiert, das auf WS-Agreement aufbaut und sich an den Arbeiten des Open Grid Forum orientiert. Ergebnis einer erfolgreichen Verhandlungsphase sind SLAs über qualitative und quantitative Attribute der zu nutzenden Ressourcen und Dienste sowie Garantien und gegebenenfalls die Einigung über mögliche Konventionalstrafen bei der Verletzung eines SLA.

Zur SLA-Provisionierung sollen Plugins für das SLA-Framework geschrieben werden, welche das SLA-Angebot des Verhandlungsinitiators übergeben bekommen. Diese Plugins sind zu einem gewissen Teil domänenspezifisch. Durch die Definition der SLA-Strukturen in AP3, ergeben sich die Eingabeparameter der Plugins. Die Anbindung an das Backend erfolgt über die Grid Middleware - UNICORE, das Globus Toolkit, beziehungsweise gLite. UNICORE verfügt bereits nativ über die notwendigen Mechanismen zur Reservierung von Ressourcen. Im Globus Toolkit soll die zur Verfügung stehende Methode genutzt werden, um wiederum mit verschiedenen Ressourcen Management Systemen zu kommunizieren. Die vorgesehenen Mechanismen zum SLA-Monitoring sind in [1] aufgeführt: Der aktuelle Status eines Agreements soll als Resource Property in Service Term States und Guarantee Term States ausgedrückt werden.

Das folgende Pseudo-Schema zeigt die Struktur eines ServiceTermStates.

```
<wsag:ServiceTermState termName="xs:string">
  <wsag:State>wsag:ServiceTermStateDefinition</wsag:State>
  {
    <wsag:Processing><xs:any##other/></wsag:Processing> |
    <wsag:Idle><xs:any##other/></wsag:Idle> |
    <xs:any##other/>
  } ?
</wsag:ServiceTermState> *
```

Der `xs:any`-Erweiterungspunkt erlaubt die Exponierung des aktuellen Status einer Reservierung. Das folgende Fragment demonstriert dies für eine Ressourcenbeschreibung, bei der der Nutzer mindestens 1.8 GHz angefordert hat und genau 2 GHz geliefert bekommt:

```
<wsag:ServiceTermState termName="JOB_DESCRIPTION">
  <wsag:State>Ready</wsag:State>
  <wsag:Processing>
    <jSDL:Resources>
      <jSDL:IndividualCPUSpeed><jSDL:Exact>2.0E9</jSDL:Exact></...>
      <jSDL:IndividualCPUCount><jSDL:Exact>2.0</jSDL:Exact></...>
      <jSDL:TotalResourceCount><jSDL:Exact>16.0</jSDL:Exact></...>
    </jSDL:Resources>
  </wsag:Processing>
</wsag:ServiceTermState>
```

Da `ServiceTermStates` als Teil des `ResourceProperty` Dokuments der `Agreement`-Instanz veröffentlicht werden, ergibt sich die Möglichkeit, über `GetResourceProperty`-Anfragen oder `WS-Notification-Subscriptions` entsprechend `WS-Agreement` den Status des `Agreements` zu beobachten. Das Monitoring im RMS hängt vom konkreten RMS ab und ist nach außen hin an dieser Stelle transparent. Die konkreten Informationen des RMS werden über einen Plugin-Mechanismus an die SLA-Schicht propagiert.

## 6 Architektur des SLA Frameworks

### 6.1 Anforderungen an die Architektur

Es wurden die folgenden Anforderungen von SLA4D-Grid bezüglich der Architektur erhoben:

- *Standard WS-Agreement-Protokoll und Möglichkeit der (Nach-)Verhandlung.*  
Aus Gründen der Interoperabilität, der Verbreitung und aufgrund des Standardisierungsstandes von `WS-Agreement` [5] ist dieses zur Repräsentierung und Erzeugung von SLAs ausgewählt worden. Es soll aus diesem Grund eine vollständig kompatible Implementierung des Protokolls vorgenommen werden. Da der Bedarf besteht, `Agreements` individuell auszuhandeln und auch nachträglich nachzuverhandeln, soll eine Erweiterung entwickelt werden, die diese Möglichkeiten bietet. Diese Erweiterung soll zunächst, unter Beteiligung des Projektes SLA4D-Grid, im Rahmen der OGF spezifiziert und dann durch das Vorhaben SLA4D-Grid umgesetzt werden.
- *Unterstützung unterschiedlicher Domänen mit eigenen Vokabularen.*  
Die SLA-Schicht muss in der Lage sein, Vokabulare aus unterschiedlichen Domänen zu unterstützen. Die `WS-Agreement`-Spezifikation ist generisch und definiert selbst keine Sprachkonstrukte, die beschreiben, welche Eigenschaften und Dienstgütern ein Dienst bieten muss. Im Rahmen des SLA4D-Grid-Projektes soll daher ein gewisser Basissatz (wie beispielsweise `JSDL` oder eine `Advanced Reservation Term Language`) unterstützt werden. Des Weiteren ist vorgesehen, dass Communities eigene Spracherweiterungen entwickeln, zu D-Grid beitragen und durch domänenspezifische Plugins realisieren.
- *Erweiterbarkeit und Revisionierung von Templates.*  
Die Erfahrung aus vergangenen Projekten hat gezeigt, dass SLAs nur sehr schwer frei spezifiziert werden können. Stattdessen basieren sie auf Templates deren Struktur sowohl Nutzer als auch Anbieter von Ressourcen bekannt ist. Da sich Anforderungen mit der Zeit ändern, soll es möglich sein, diese Templates nachträglich anzupassen und zu revidieren.
- *Komponierbarkeit (z.B. Komponente für Auswerten einer JSDL Beschreibung).*  
Da unterschiedliche Domänen häufig gemeinsame Anforderungen besitzen wie zum Beispiel die Beschreibung der angeforderten Infrastruktur (zum Beispiel mittels `JSDL`) oder Zeitpunkte zu denen eine Reservierung durchgeführt werden soll, muss es möglich sein, diese Aspekte in separaten Komponenten zu kapseln und für domänenspezifische SLAs zu komponieren.
- *Unterstützung von D-Grid Middlewares.*  
Da das `Globus Toolkit 4.0` auf einer Draft-Version der `WS-Resource Framework` und `WS-Addressing` Spezifikationen basiert, während `UNICORE` die finalen Standards verwendet, wird das `Globus Toolkit 4.2` voraussichtlich aus

Interoperabilitätsgründen als Umgebung für die Globus Welt fungieren. Jedoch ist zu berücksichtigen, dass die Entscheidung über die zu verwendende Globus Version von den Anforderungen des D-Grid abhängt, daher ist diese Entscheidung noch nicht endgültig. Was gLite betrifft, so ist zwar eine mögliche Architektur zur Integration des generischen Anwendungsfalles beschreiben, eine konkrete Umsetzung und ihre Implikationen liegen allerdings noch nicht vor.

- *Skalierbarkeit.*  
Die Skalierbarkeit der SLA-Schicht soll auf zwei Wegen gewährleistet werden. Zum einen soll es möglich sein, parallel Verhandlungen von SLAs durchzuführen. Zum anderen soll die Aushandlung von Rahmenverträgen erlauben, dass für eine große Zahl kleiner Aufträge keine individuellen Verhandlungen durchgeführt werden müssen.
- *Überprüfbarkeit von Agreements.*  
Da Service Level Agreements Provisionen und Strafen für die Erfüllung bzw. Verletzung der beschriebenen Dienstgütern definieren, ist es wichtig, dass die realisierten Dienstgütern nach außen überprüfbar dargestellt werden, sodass eine Abrechnung erfolgen kann.
- *Anforderungen bezüglich D-Grid Integration*  
Die SLA-Schicht soll alle notwendigen Voraussetzungen mitbringen, sodass eine Anbindung an Ressource-Management-Systeme, Billing-Systeme und VO-Management-Systeme erfolgen kann.

## 6.2 Umsetzung der Architektur

Abbildung 4 zeigt die Architektur der SLA-Schicht Stand Projektmonat 4, die einzelnen Elemente werden im Folgenden beschrieben.

### 6.2.1 Komponente der SLA Preselection

#### SLA Template Registry (Discovery)

Die SLA Template Discovery Komponente kommuniziert mit dem agreementFactoryPortType und der Discovery Komponente der Community. Hierdurch kann ein geeignetes SLA Template aus der SLA Template Registry aus dem Bereich SLA-Verhandlung und -Überwachung genutzt werden, dass den Ansprüchen des Specific SLA Client gerecht wird.

Hierbei greift die SLA Template Registry (Discovery) auf sämtliche notwendigen im D-Grid vorhandenen Daten zu. Weiterhin werden die SLAs der verschiedenen Standorte im D-Grid ermittelt, um ein angemessenes SLA zu identifizieren.

#### User

Der User ist in der Lage, Anfragen an die SLA Template Registry (Discovery) zu stellen. Nach eingegangener Anfrage des Users identifiziert sich die SLA Template Registry (Discovery).

#### Admin

Der Admin hat die Berechtigung die SLA Template Registry (Discovery) zu aktualisieren. Dies geschieht in Projektmonat 12 zunächst manuell.

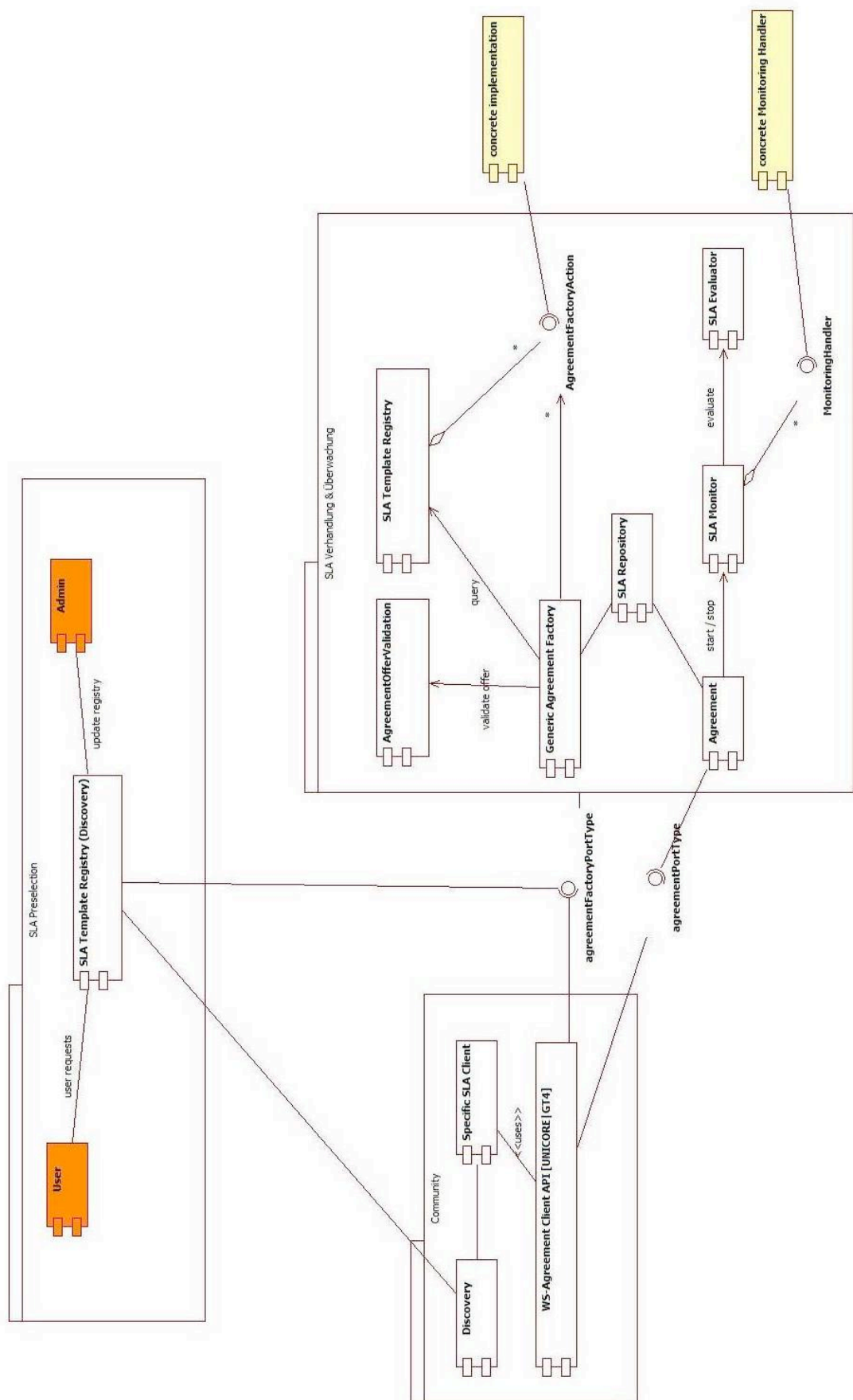


Abbildung 4 Die Architektur der SLA-Schicht (Stand Projektmonat 04)

## 6.2.2 Komponenten der Community

### Discovery

Die Discovery Komponente der Community ist der Mediator zwischen SLA Template Registry (Discovery) und Specific SLA-Client. Die Discovery ermittelt einen geeigneten Dienst.

### Specific SLA-Client

Beim Specific SLA-Client handelt es sich um den Client eines spezifischen Dienstes, der die domänenspezifischen Anforderungen des Dienstes berücksichtigt. Seine Aufgabe ist es, den Benutzer bei der Aushandlung von SLAs zu unterstützen. Es werden Specific SLA-Clients für die beiden Anwendungsfälle in SLA4D-Grid, den von der con terra GmbH sowie den von Sun Microsystems entwickelt, die zugleich als Beispiel für weitere Community-Projekte fungieren. Aufgrund der Komponierbarkeit können die dort unterstützten Aspekte in weitere Specific SLA-Clients übernommen werden. Der Specific SLA-Client nutzt das WS-Agreement Client API zur Kommunikation mit der SLA-Implementierung der Middleware. Hierbei stellen Specific SLA-Clients eine konkrete Schnittstelle zum Benutzer dar.

### WS-Agreement Client API

Die WS Agreement Client API ist ein Interface, das mittels des WS-Agreement-Protokolls mit einem WS-Agreement-Service kommuniziert. Hierbei finden der AgreementFactoryPortType und der AgreementPortType Berücksichtigung. Das WS-Agreement Client API stellt die clientseitige Implementierung des WS-Agreement-Protokolls dar und ist agnostisch hinsichtlich spezifischer SLAs, ihrer Struktur und ihres Inhalts. Das API stellt die benötigten Funktionalitäten bereit, um über die in der WS-Agreement-Spezifikation definierten Mechanismen (Port Types) mit WS-Agreement konformen Diensten zu kommunizieren. Die WS-Agreement-Spezifikation umfasst die folgenden Port Types: AgreementFactory, PendingAgreementFactory, Agreement-Acceptance, Agreement und AgreementState. Das WS-Agreement Client API soll Middleware-Aspekte wie Authentifizierung und sichere Kommunikation dem Specific SLA-Client gegenüber transparent machen.

Momentan sind zwei Implementierungsmöglichkeiten der WS-Agreement Client API vorgesehen, wobei sich die zweite aus der ersten heraus entwickelt:

- a) Die Client API umfasst einen Layer für jede Middleware, also jeweils einen für gLite, einen für das Globus Toolkit und einen für UNICORE (siehe Abbildung 5). Diese drei Layer existieren parallel und kommunizieren über eine Facade mit dem Klienten. Diese API wird mit dem ersten Prototypen in Projektmonat 12 ausgeliefert.

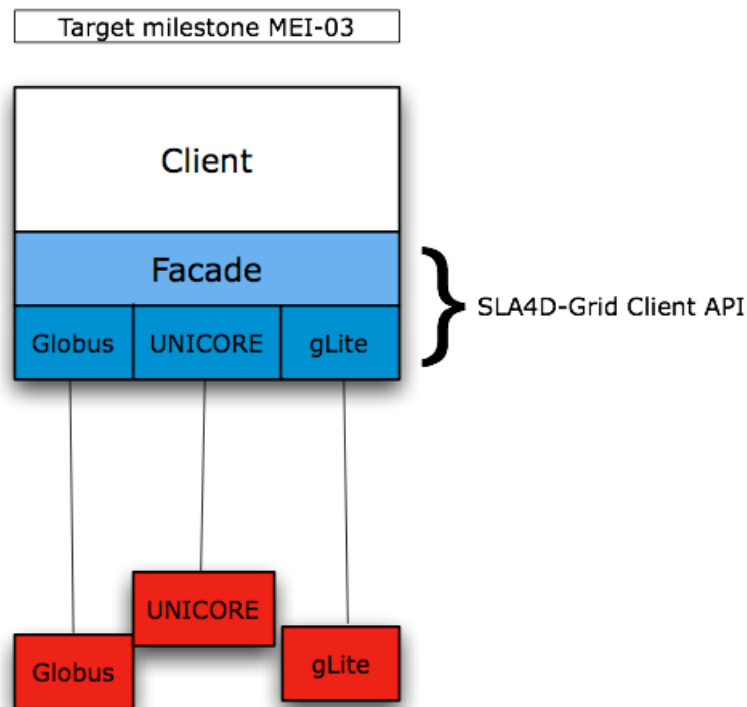


Abbildung 5 SLA4D-Grid Client API Prototyp I (MEI-03, Projektmonat 12)

- b) Die Client API des zweiten Prototypen der SLA-Schicht trennt die Facade und die spezifischen Implementierungen für jede der drei Middlewares ab und implementiert das WS-Agreement Protokoll zwischen Klient und SLA-Schicht (siehe auch Abbildung 6). Diese Implementierung ermöglicht größtmögliche Freiheiten bei der Umsetzung community-spezifischer Anwendungsfälle und wird auch mit der finalen Version der SLA-Schicht ausgeliefert werden.

Das zweistufige Modell ist gewählt worden, da die Client API gemäß Abbildung 6 einige Vorarbeiten an der WS-Agreement Implementierung erfordert, die die Auslieferung des ersten Prototypen verzögern könnten. Daher ist entschieden worden, zuerst gemäß Abbildung 5 zu implementieren, um die Auslieferung von Prototyp I gemäß dem Zeitplan zu gewährleisten. Es sei angemerkt, dass der Aufwand für die Umstellung von einer auf die andere API gering sein wird.

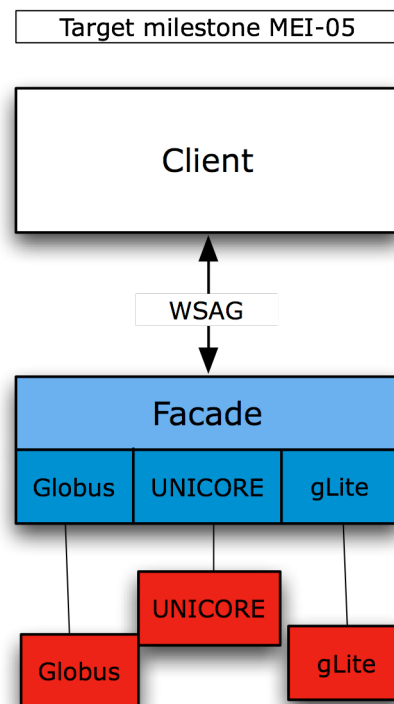


Abbildung 6 SLA4D-Grid Client API Prototyp II (MEI-06, Projektmonat 24)

### 6.2.3 Komponente für SLA-Verhandlung und -Überwachung Generic Agreement Factory

Die Generic Agreement Factory ist eine Kernkomponente der SLA-Schicht und repräsentiert eine generische Implementierung eines WS-Agreement Factory Web Services. Dieser ist für die Kommunikation mit dem WS-Agreement Client API zur Erzeugung von SLAs zuständig. Er orchestriert die Nutzung diverser weiterer Komponenten, welche die Verhandlung sowie den vollen Lebenszyklus eines SLAs kontrollieren. Hierbei implementiert der SLA-Layer den AgreementFactory-PortType und den PendingAgreementFactory-PortType.

Der AgreementFactory-PortType erlaubt Webservice-Clients die synchrone Erstellung von Agreements, während der PendingAgreementFactory-PortType die asynchrone Erstellung von Agreements ermöglicht. Sowohl Anfragen an den AgreementFactory-PortType als auch an den PendingAgreementFactory-PortType werden an diese generische Agreement Factory delegiert.

Die generische Agreement Factory realisiert die Verbindung zwischen der Webservice-Schicht und der SLA-Schicht. Sie lädt Agreement Templates aus der Template Registry und erzeugt neue Agreement-Instanzen basierend auf den eingehenden Anfragen (Agreement Offer). Jede Anfrage muss auf einem Template der Agreement Factory basieren. Die generische Agreement Factory validiert eingehende Anfragen zur Erstellung neuer

Agreements hinsichtlich der Creation Constraints, die in dem zugrunde liegenden Template definiert sind. Damit wird sichergestellt, dass Anfragen zur Erstellung Agreements korrekt von einem Agreement Client erzeugt wurden. Die generische Agreement Factory nutzt die Agreement Offer Validation zum Prüfen der Gültigkeit einer Anfrage.

Nach der erfolgreichen Validierung einer Anfrage führt die generische Agreement Factory die spezifische Logik zur Erstellung des Agreements beziehungsweise zur Erzeugung des mit dem Agreement verbundenen Dienstes aus. Im Falle einer erfolgreichen Erzeugung des Dienstes wird eine neue Agreement-Instanz erzeugt.

### AgreementOfferValidation

Die AgreementOfferValidation-Komponente ist damit beauftragt, vor dem Akzeptieren eines SLA-Angebots zu prüfen, ob dieses den CreationConstraints des zugehörigen SLA-Templates genügt. Hierbei unterstützt die WS-Agreement-Spezifikation die Definition so genannter „Creation Constraints“ als Teil eines Agreement Templates. Sie definieren sowohl die Struktur als auch die Wertebereiche einzelner Elemente eines Templates. Eine gültige Anfrage zur Erstellung eines Agreements muss auch hinsichtlich der Creation Constraints des ihr zugrunde liegenden Agreement Templates gültig sein. Der Agreement Offer Validator überprüft die Validität jeder Creation Constraint für eine eingehende Anfrage. Dazu generiert der Validator für jede Creation Constraint einen XML Schema-Datentyp basierend auf der Constraint. Anschließend selektiert er die Daten der Anfrage, die durch die jeweilige Creation Constraint referenziert werden. Im Anschluss werden die selektierten Daten mittels XML Schema-Validierung hinsichtlich ihrer Gültigkeit entsprechend dem Constraint-Datentyp überprüft. Mittels der AgreementOfferValidation-Komponente kann bereits über das Design des SLA-Templates verifiziert werden, dass nur solche Ressourcen angefordert werden können, die auch tatsächlich angeboten werden können. Die AgreementOfferValidation stellt eine Art Vorfilter für eingehende Anfragen dar.

### SLA Template Registry

Die SLA Template Registry speichert SLA-Templates, die von Nutzern angefordert werden können und als Grundlage für neue SLAs dienen. Die Agreement Factory nutzt bei Anfragen die im SLA Template Repository gespeicherten Templates. Das SLA Template Repository soll in der Lage sein, Templates zu revisionieren. Des Weiteren soll es dazu dienen domänenspezifische Plugins an bestimmte Templates und deren Revisionen zu binden, so dass die domänenspezifischen Anforderungen realisiert werden können.

Weiterhin beinhaltet die Agreement Factory Action-Mechanismen zur Verhandlung von Agreement-Templates und zur Erzeugung von neuen Agreements. Die Template Registry ermöglicht einer Agreement Factory den Zugriff auf Agreement Templates und den zugehörigen Factory Actions über den Namen und die ID eines Templates. Zudem ermöglicht sie das Auflisten aller Templates dieser Registry.

## SLA-Repository

Im SLA-Repository werden abgeschlossene SLAs mit allen assoziierten Daten gespeichert. Hierdurch ist es möglich, auf diese sowohl zur Laufzeit als auch später zuzugreifen. Dies ist unter anderem für das Accounting und Billing relevant. Das SLA-Repository ist primär für die statischen Daten des SLAs zuständig, da die dynamischen Daten dem Monitoring entstammen.

Das SLA Repository stellt die Persistenzschicht der SLA-Schicht dar.

## Agreement

Eine Agreement-Instanz speichert die Daten (Properties) eines Agreements und implementiert die Geschäftslogik zum terminieren des Agreements. Zu den Daten des Agreements zählen neben dem Agreement Context und den Agreement Terms unter anderem auch die verschiedenen Zustände einer Agreement-Instanz. Die Zustandsdaten des Agreements werden während des SLA-Monitoring-Prozesses regelmäßig aktualisiert. Zu den Zustandsdaten gehören der Agreement-Status, der Service Term-Status und der Garantie Term-Status. Agreement-Clients können die Daten mittels der in WSRF spezifizierten GetResourceProperty-Methode abfragen. Hierdurch ist der Agreement-Webservice mit dem SLA-Repository sowie dem Monitoring verbunden und kommuniziert mit dem Nutzer über das WS-Agreement Client API.

## SLA-Monitor

Der SLA-Monitor sammelt die relevanten dynamischen Daten eines SLAs während der Ausführung. Die Zustandsdaten eines Agreement bestehen aus dem Agreement-Status, dem Service Term-Status und dem Garantie Term-Status. Der SLA-Monitor aktualisiert diese Zustandsdaten periodisch. Der Monitor wird beendet sobald das Agreement einen finalen Status erreicht.

Der Agreement-Status repräsentiert den Zustand des SLA-Monitoring-Prozesses. Solange sich ein Agreement in einem überwachten Status befindet wird der Status der Service Terms und der im Agreement definierten Garantien aktualisiert.

Für jeden Service Term eines Agreements existiert ein Service Term-Status. Dieser Status spezifiziert den Zustand eines Service Terms (NotReady, Ready, Complete). Darüber hinaus kann dieser Status zusätzliche Informationen zu einem Service Term beinhalten, die zur Auswertung der Garantien des Agreements notwendig sind.

Für jede im Agreement definierte Garantie existiert ein Garantie Term-Status. Dieser Garantie Term-Status wird auf Basis der Service Term-Statusobjekte ermittelt. Der SLA-Evaluator ist für die Auswertung des Garantie Term-Status verantwortlich. Hingegen nimmt der SLA-Monitor keine Auswertung vor.

## SLA-Evaluator

Der SLA-Evaluator bezieht zur Laufzeit Daten vom SLA-Monitor und führt eine Auswertung durch. Hierbei ermittelt er auf Basis der Agreement Terms und der Service Terms den Status der einzelnen Garantien (Guarantee Terms) des Agreements. Der SLA-Evaluator wird zu diesem Zweck vom SLA Monitor aufgerufen, nachdem die Zustände der Service Terms vom Monitor aktualisiert wurden.

Der SLA-Evaluator stellt zudem die Schnittstelle zum SLA-Accounting dar. Er stößt die Meldung von SLA-Verletzungen und -Erfüllungen für das externe Monitoring an und bietet dem Agreement Webservice die Daten für die Garantien. Die Auswertung der Garantien resultiert in Gutschriften und Strafen, basierend auf der Erfüllung bzw. Verletzung der definierten Garantien.

#### 6.2.4 Konkrete Implementierungen

Um die Unterstützung von verschiedenen Domänen zu ermöglichen, ist es notwendig, dass Plugins domänenspezifische Funktionalitäten übernehmen. Beispiele für solche Plugins sind Entscheidung, ob ein SLA-Angebot akzeptiert werden kann, die Durchführung der Allokation von Ressourcen im RMS, das Monitoring, etc.

##### AgreementFactoryAction

Hierbei handelt es sich um die konkreten Implementierungen der Logik einer Agreement Factory. Die AgreementFactoryAction realisiert die Geschäftslogik zur Verhandlung von Templates sowie zur Erstellung neuer Agreements. Die Aktionen erlauben Entscheidungen zu fällen, ob SLA-Angebote akzeptiert werden, und kommunizieren mit dem zugrundeliegenden RMS. Da eine Validierung von SLA-Angeboten gegen die SLA-Templates bereits zuvor von der AgreementOfferValidation-Komponente durchgeführt wurde, können diese domänenspezifischen Aktionen von konkreten SLA-Strukturen ausgehen und sehr einfach relevante Daten aus dem SLA-Angebot extrahieren und umsetzen. Die Aktionen selbst zerlegen sich in komponierbare Teilaspekte, sodass zum Beispiel die Allokation von Ressourcen zwischen verschiedenen domänenspezifischen Implementierungen geteilt wird, während das Starten und Beenden von domänenspezifischen Diensten in diesen Ressourcen individuell implementiert wird.

##### Monitoring Handler

Ein oder mehrere Monitoring Handler werden mit einem konkreten Agreement assoziiert. Die Monitoring Handler sind für das Monitoring erforderlich und liefern die Daten anhand derer entschieden wird, ob die Garantien eines SLAs erfüllt oder verletzt werden. Diese Daten können aus dem RMS direkt entstammen oder Monitoringdiensten des D-Grid entnommen werden.

Die Monitoring Handler sind verantwortlich für die Aktualisierung der Zustände der Service Terms und realisieren die Verbindung zum Service Monitoring.

## 7 Arbeitsplan bis Projektmonat 12

In Projektmonat 12 wird ein erster Prototyp der SLA-Schicht vorliegen (korrespondierend mit Meilenstein MEI-03). Aus Sicht von Arbeitspaket 2 wird hier insbesondere angestrebt, dass ein SLA-Client mit der AgreementFactory sowie Agreement Services in den Grid Middlewares kommunizieren kann. Dies bedeutet insbesondere auf Seiten der Globus und der gLite Middleware eine bedeutende Entwicklung und Integrationsleistung. Dazu muss die zugrundeliegende WS-Agreement Implementierung, voraussichtlich WSAG4J<sup>8</sup>, angepasst und um den SOAP Stack reduziert werden.

Weiterhin ist die Umsetzung des generischen D-Grid SLAs und des entsprechenden Anwendungsfalles, der generischen Reservierung von Ressourcen für Rechenjobs vorgesehen. Zudem wird die Reservierung von Speicher mittels gLite wird analysiert und die Möglichkeiten einer Integration eines Netzwerkreservierungssystems wird untersucht.

An dieser Stelle ist zu bemerken, dass eine einheitliche Umsetzung aller in Abschnitt 6 vorgestellter Komponenten und Dienste über alle drei Middlewares weder vorgesehen noch realisierbar ist. Insbesondere ist der generelle Entwicklungsstand der SLA-bezogenen Arbeiten für gLite (auch unabhängig von SLA4D-Grid) noch rudimentär, so dass hier an vielen Stellen bei Null anzufangen ist.

### 7.1 Aktueller Stand der Entwicklung

Die bereits vorhandenen Komponenten sind im Folgenden aufgelistet:

- Generische Komponente eines Verhandlungsdienstes:  
Eine Anpassung der Komponente ist für die verschiedenen Anwendungsfälle erforderlich.
- Eine Komponente, die die Aufgaben der Agreement Factory erfüllt.
- Ein SLA Template Speicher.
- Client API

Die oben genannten Komponenten wurden im Rahmen diverser Projekte entwickelt und sind integriert als generisches SLA Framework WSAG4J verfügbar. Jedoch ist zu berücksichtigen, dass sämtliche Komponenten angepasst werden, um eine optimale Lösung für die angestrebten Ziele zu liefern.

Zudem existieren Komponenten, die Teile der von SLA4D-Grid vorgegeben Anforderungen bereits erfüllen, beziehungsweise dies nur für eine der vorgesehen Middlewares tun:

- Eine Provisionierungs-Komponente für UNICORE:  
Die entsprechende Komponente ist in UNICORE implementiert. Weiterhin ist eine grundlegende Anpassung der Komponente erforderlich.  
Eine detaillierte Aufwandsabschätzung wird hier noch erfolgen.

<sup>8</sup> <http://packcs-e0.scai.fraunhofer.de/wsag4j/index.html>

- Eine Provisionierungs-Komponente für Globus Toolkit 4.  
Die vorläufige Aufwandsabschätzung zur Anpassung der Komponenten beträgt 2 Monate.
- Eine Provisionierungs-Komponente in gLite.  
Eine detaillierte Aufwandsabschätzung wird hier noch erfolgen.
- Eine Provisionierungs-Komponente für MPLS- und GMPLS-Netzwerke.  
Eine detaillierte Aufwandsabschätzung wird hier noch erfolgen.
- Eine generische Monitoring-Komponente:  
Eine detaillierte Aufwandsabschätzung wird hier noch erfolgen.

## 7.2 Ziele für Projektmonat 12 (Meilenstein MEI-03)

Die folgenden Komponenten sollen bis Projektmonat 12 erreicht werden:

- Umsetzung eines generischen D-Grid-SLAs. Voraussetzungen hierfür sind:
  1. Abstrahierung des SOAP Stacks aus der WS-Agreement Implementierung. Die vorläufige Aufwandsabschätzung hierfür beträgt einen Monat.
  2. Die Umsetzung fehlender respektive nur teilweise vorhandener Kerndienste und der Provisionierung.

Folgende Abhängigkeiten bestehen hierbei zu

- der Spezifikation des generischen SLAs (AP3, SLA4D-Grid).
- der Anbindung an die Sicherheitsinfrastruktur im D-Grid (direkte Abhängigkeit zum D-Grid).

Offene Punkte:

- Eine D-Grid-weite Discovery.

## 8 Anmerkungen

Das vorliegende Dokument beschreibt eine erste Version Architektur der SLA-Schicht für das D-Grid. Diese Architektur wurde basierend auf ausgewerteten Anwendungsfällen, Rückmeldungen der Community, Integrationsaspekten bezüglich des D-Grid und basierend auf Erfahrungen mit vorhergehenden Projekten entwickelt. Das Projekt SLA4D-Grid und die Verantwortlichen in Arbeitspaket 2 gehen davon aus, dass die Kernarchitektur über die Projektlaufzeit größtenteils erhalten bleiben wird und folgende Versionen primär Erweiterungen und Verbesserungen mit sich bringen werden. Daher werden die zukünftigen Projektberichte von AP 2 Erweiterungen des vorliegenden Dokumentes sein, so dass dieses „Living Document“ zu verstehen ist.

## Referenzen

- [1] B. Baranski, D. Battré, V. Keller, and W. Ziegler, *Initial Version of D-Grid SLA*, SLA4D-Grid Projektbericht D3.1, TR-SLA4DGRID-D3.1, Oktober, 2009.
- [2] D. Mallmann, *Anforderungen und existierende Lösungen im IT-Serviceumfeld für SLAs*, Projektbericht D6.1, TR-SLA4DGRID-D6.1, September, 2009.
- [3] The Highly-Available Resource Co-allocator: <http://www.cct.lsu.edu/harc.php>
- [4] Barz, C. and Bornhauser, U. and Martini, P. and Pilz, M. and de Waal, C., Willner, A., *ARGON: Reservation in Gridenabled Networks*, Proceedings of the 1. DFN-Forum on Communication Technologies 2008.
- [5] Andrieux, A.; Czajkowski, K.; Dan, A.; Keahey, K.; Ludwig, H.; Nakata, T.; Pruyne, J.; Rofrano, J.; Tuecke, S.; Xu, M. *Web Services Agreement Specification (WS-Agreement)*, Grid Forum Document GFD.107; The Open Grid Forum, Joliet, Illinois, United States, 2007.
- [6] Battré, D., Havestadt, M., Wäldrich, O., *Lessons learned from implementing WS-Agreement*, Grid 2009, IEEE Workshop on Service Level Agreements in Grids, Springer, CoreGRID Series, erscheint Sommer 2010.