GFD-I.141
ISV 向けリモートコンピューティング利用入門

著者：
Steven Newhouse（マイクロソフト）
Andrew Grimshaw（バージニア大学）

2008 年 10 月 7 日

独立系ソフトウエアベンダー（ISV）向けリモートコンピューティング利用入門

本文書の位置づけ
本文書は、リモートコンピューティングシナリオをサポートする際に、オープングリッドフォーラムや他の標準化団体（SSO）の仕様をどのように使うかということについて、独立系ソフトウエアベンダー（ISV）に情報を提供するものである。本文書はいかなる標準も定義せず、技術的な推奨を行うことはない。配布は自由である。

商標
OGSA は登録商標であり、オープングリッドフォーラムのサービスマークである。

要旨
本文書では、分散した計算リソースへデスクトップからアクセスするため、オープングリッドフォーラムの内外で作成されてきた仕様をどのように利用するのかを紹介する。独立系ソフトウエアベンダー（ISV）は、コンポーネントが 2 つあるアプリケーションを開発することが多くなってきた。2 つのコンポーネントとは、アプリケーション上のグラフィカルクライアントと、「バックエンド」の計算クラスタで走らせることのできる数値計算サーバコンポーネントである。これら 2 つのコンポーネントを、ネットワーク内で統合するには多くの問題がある。ネットワークにはファイアウォールや NAT が設定されていることが多く、異なる OS やソフトウエア環境を超えて統合することになるからである。この環境への標準に準拠したインタフェースがあれば、ISV にアプリケーション構築のための確固たる基盤が与えられることになる。

分散した計算リソースへのアクセスを促進するために開発されてきた一連の標準や仕様を

紹介し、解説する。これらの仕様が分散した計算リソースへのアクセスをどのように可能にするのかを、5 つのシナリオを通じて説明する。クライアントとクラスタが共通のファイルシステムをもっているような計算クラスタに対して単純にジョブを投入するというシナリオから、計算クラスタ上で走っているアプリケーションとリモートクライアントとの間で双方向対話を行いながらクライアントが計算クラスタへ（計算クラスタから）ファイルをステージングするシナリオまで、さまざまである。

これらのシナリオに仕様がどのように用いられるのかを示し、ミドルウェアプロバイダと ISV の両者に対する要件を提示する。ただしこれは厳格で規範的な解決方法を示すものではなく、むしろアドバイスと解釈すべきものである。最後に、リソースの選択や安全性に関係する団体が議論やフィードバックを行うための、未解決の問題をいくつか指摘する。

目次

1. はじめに

多くのユーザは、自分たちのクライアントワークステーション上にある ISV 提供のデスクトップアプリケーションと、「バックエンド」の計算クラスタとを、一律の方法あるいは標準的な方法で統合したいと考えている。これには接続済みのクライアントワークステーションも未接続のクライアントワークステーションも含まれることがある。また、クライアントワークステーション、バックエンドの計算機、サードパーティのデータサーバのそれぞれの間でファイルを移動する作業も含まれることがある。さらに、ファイアウォールや NAT あるいは現実世界に存在するその他のネットワーク上の障害などから発生する問題もある。いいかえれば、クライアントワークステーションおよび（または）バックエンドの計算機が、グローバルにアドレス可能な IP アドレスをもたないことがあるのである。

本文書では、分散した計算リソースへデスクトップからアクセスするため、オープングリッドフォーラムの内外で作成されてきた仕様をどのように利用するのかを紹介する。これは、これらの仕様の利用法について提言をするものであって、厳格かつ規範的な解決方法を示すものではない。分散した計算リソースやファイルストレージリソースをもつ一連の典型的な計算環境を明示し、そうした環境の中で仕様を使用する典型的なシナリオを紹介する。これらのシナリオについて、関連する仕様を特定し、その使い方を示す。とくに、共有ファイルシステムへのステージングの代わりに、計算リソースへのファイルのステージングや計算リソースからクライアントワークステーションへのファイルのステージングを行うことのあるアプリケーションの実行、さらにこれに加えてネットワークのファイアウォールを通じてクライアントワークステーションが実行中のアプリケーションと双方向対話を行うことのあるようなアプリケーションの実行を解説する。実装や採用を明確に推奨する上で、関連団体とさらに議論を深める必要があると我々が感じている問題点を最後に提示する。

一般の物理的インフラストラクチャを以下に示す。

図 1　一般の物理的インフラストラクチャ
Submission Host サブミッションホスト
Client Workstation クライアントワークステーション
Client Firewall クライアントファイアウォール
Enterprise Firewall　社内ファイアウォール
License Manager　ライセンスマネジャ
System Manager　システムマネジャ
System Monitoring　システム監視
Cluster Firewall　クラスタファイアウォール
Head node　ヘッドノード
Compute nodes　計算ノード

1.1 主要機器
図に描かれた機器が持つ重要な特徴をここに定義しておく。

・クライアントワークステーション：エンドユーザがジョブの投入、監視、結果の閲覧などを行うマシン。クライアントマシンは、可動式でネットワークからしばしば切り離すことがあり、社内ネットワークや計算リソースとファイルシステムを常時共有するものではないと考えてよい。
・サブミッションホスト：ユーザが計算クラスタにジョブを投入したり監視したりすることのできるワークステーション。一般に、社内ネットワークに常時接続し、会社やクラスタのファイルシステムにアクセスすることのできるマシンを指す。
・バックエンドの計算クラスタ：以下の機器から構成される。
〇ヘッドノード：サブミッションホストがジョブをクラスタに投入する際に、窓口となるところ。おそらく計算クラスタの中で、他のネットワーククライアントから見える唯一の部分である。
〇計算ノード：ヘッドノードが管理する計算クラスタ。計算ノードは、ヘッドのノードからのみアクセス可能とすることができる。
〇ヘッドノードと計算ノードに共通の「クラスタ」ファイルシステム。このシステムは、サブミッションホストからはアクセスできるが、クライアントワークステーションからはアクセスできない。
・外部ストレージサーバ：社内ネットワークの外側（クライアントワークステーション）や内側（サブミッションホスト）あるいはクラスタのヘッドノードから、認証可能なネットワークプロトコル（http、scp、ftp 等）を通じてアクセスできるストレージリソース。必ずしもグローバルなファイルストレージシステムである必要はない。
・ネットワークルータ：ワークステーション、サブミッションホスト、ヘッドノードをつ

なぐもの。
・ファイアウォール：ヘッドノード、クライアントワークステーション、サブミッションホストを保護するものの一つ。

1.2 シナリオ
本文書で考えるシナリオを簡単に説明しておく。これらのますます複雑になるシナリオにおいては、ネットワーク環境の複雑さが主な差となっている。それぞれ基本となるシナリオは、デスクトップ環境から計算クラスタ上でアプリケーションを実行することにある。
1. 専用の分散リソースマネジャを直接使用して、サブミッションホストからクラスタ上でアプリケーションを実行する。
2. 標準化 API を用いて、サブミッションホストからクラスタ上でアプリケーションを実行する。
3. ウェブサービスプロトコルと共有ファイルシステムを用いて、サブミッションホストからクラスタ上でアプリケーションを実行する。
4. ウェブサービスプロトコルと、共有ファイルシステムの代わりにファイルステージング（シナリオ 3 との主な相違点）とを用いて、クライアントワークステーションからクラスタ上でアプリケーションを実行する。
5. ウェブサービスプロトコルを用いて、クライアントワークステーションからクラスタ上でアプリケーションを実行する。ただしその際、クライアントワークステーションと実行中のアプリケーションとの間に双方向対話があるものとする。

それぞれのシナリオに対し、環境、前提条件、機器間の対話の詳細、シナリオから得られるメリットを示していく。これらのシナリオを以下にまとめる。

	シナリオ				
特徴	1	2	3	4	5
アプリケーションが専用のクライアント API を使用する。	はい	いいえ	いいえ	いいえ	いいえ
アプリケーションが DRM のクライアントライブラリのインストールを必要とする。	はい	はい	いいえ	いいえ	いいえ
ウェブサービスプロトコルが、クライアントとクラスタをリンクしている。	いいえ	いいえ	はい	はい	はい
クライアントとクラスタに共有ファイルシステムが必要である。	はい	はい	はい	いいえ	いいえ
ファイル転送プロトコルが、クライアン	いいえ	いいえ	いいえ	はい	はい

トとクラスタをリンクする。					
ネットワークからの切断時に作業を行うため、クライアントからファイルを明示的に転送する	いいえ	いいえ	いいえ	はい	はい
クライアントとクラスタ上で走っているアプリケーションとの間のインタラクティブな双方向通信がある。	いいえ	いいえ	いいえ	いいえ	はい

図 2　各シナリオが持つ特徴

シナリオ 3、4、5 はジョブを投入するためにウェブサービスを使うという共通の特徴がある。シナリオ 4、5 の特徴は一つにまとめることができるが、ここでは簡単にするため切り離している。

1.3 付加条件

このシナリオの目的は、クライアントワークステーションが「難しい」ネットワーク環境にある場合でも、「良い」ユーザ体験が得られるようにすることにある。具体的には以下の通りである。

・多くの計算アプリケーションが、計算ノードと対話を行う要素を持っている。そのようなシナリオでは、計算ノードは、中間結果を出すためにクライアントワークステーションに「接続し返す（コネクトバックする)」ことができる必要がある。
・いかなるソリューションも、クライアントワークステーションや企業内で使用できる既存のセキュリティメカニズムを利用しなければならない。ISV のアプリケーションに特注のセキュリティソリューションを埋め込むべきではない。

1.4 現在のところ想定していない事項

これらのシナリオでは、現時点で有用なソリューションを与えるため、現実的な簡単化を行っている。特に以下のような簡単化をしている。

・クライアントワークステーション上で走っているクライアントアプリケーションからは、計算クラスタが見えないことを想定しており、これが見えてしまうようなプロセスは考慮していない。ある組織内で「よく理解されている」リソースを想定し、さらにあるリソースが他のリソースをカプセル化することがあるといったケースを想定しているが、その場合、こうした動作はクライアントアプリケーションからは見ることができない。
・シナリオの中でアプリケーションを使用することについては説明しない。クラスタの管理システムがアカウンティング機能を持っていることがよくあるが、これらのシナリオの

中では明記しない。
・ライセンスマネジャについては考慮に入れていない。アプリケーションがライセンスマネジャの存在を必要とすることはよくあるが、適切なライセンスが得られるまでクラスタ上でのアプリケーションの実行が遅れることがある。そのような状況では、クラスタ管理システムが示す状態にこうした遅延が反映される場合がある。

2. さまざまな仕様

クライアントと、クラスタ上にインストールされたアプリケーションとの間の対話を可能にする際、オープングリッドフォーラムやその他の標準団体が作成したウェブサービスや関連仕様を使用する。これらを以下に簡単に説明する。仕様の詳しい情報や技術的詳細については、他の文献に譲る。

2.1 WS アドレッシング

WS アドレッシング（WS-Addressing）の仕様は、エンドポイントリファレンス（EPR）と呼ばれる拡張可能な XML データ構造を定義する。EPR は、クライアントがサービスを送信するのに必要とする情報をカプセル化する働きをする。EPR には、ネットワークプロトコルアドレス、セキュリティポリシなどの提案事項を伝える拡張可能なメタデータセクション、セッション識別子やリソース識別子等のための外からは見えないセクションなど、さまざまなデータが含まれる。

2.2 セキュリティ仕様とプロファイル

ウェブサービスセキュリティ（WS-Security）関連の仕様は、セキュリティクレデンシャルをメッセージコンテンツに関連づけるための多目的メカニズムを定義する。このメッセージコンテンツは、広く使われている種類のトークン（たとえば X.509、Kerberos、SAML、ユーザ名トークンなどのクレデンシャル）をエンコードするための OGF 特有のプロファイルを構成するのに使われる。WS セキュリティコア(WS-Security Core)仕様は、背景にある通信プロトコルの助けを借りずにエンド・ツー・エンドのメッセージングの整合性や機密性を保つため、XML エンクリプションや XML デジタル署名の使用を定義するものである。現実の相互運用性を実現するため、WS-I BSP （ベーシックセキュリティプロファイル）は、WS セキュリティおよび関連するセキュリティトークンのフォーマットを使用する際の指針を与えている。これは通信を行う実装間の微妙な差異やあいまいさをなくすためであり、共通のセキュリティメカニズムを利用することを目指している。

WS セキュリティで定義されているプロファイルの中で関連するものは 2 つある。WS セキュアアドレッシング（WS-Secure Addressing）（GFD131）と WS セキュアコミュニケーション（WS-Secure Communication）（GFD132）である。WS セキュアアドレッシングは、

WS アドレッシングと WS セキュリティポリシ（WS-Security Policy）上のプロファイルであり、キー、タイプ、アイデンティティ、セキュリティポリシなどのメタデータを安全にバインドしたものを WS アドレッシングのエンドポイントリファレンス（EPR）にアドレスする。たとえば、エンドポイントはどのような認証メカニズムをサポートあるいは要求するか、といったものである。WS セキュアコミュニケーションは、WS-I ベーシックセキュリティプロファイルで定義された一般的なセキュリティメカニズムをプロファイルする。

2.3 ジョブサブミッション記述言語(JSDL)（GFD56）
JSDL は、アプリケーション、アプリケーションに必要なリソース（メモリ、CPU の数や速度等)、アプリケーションが実行される前にステージインするファイル、アプリケーションの終了時にステージアウトするファイル、実行するコマンドライン文字列などを記述するための XML ベースのスキーマである。JSDL はジョブを記述するための用語を定義し、新たな用語の定義もこれを用いて行うことができる。

2.4 JSDL シングルプロセス・マルチデータ・アプリケーションエクステンション（GFD115）
SPMD アプリケーションエクステンションは、並列アプリケーションの定義をサポートするため、JSDL アプリケーションエレメントへの多くの追加事項を定義するものである。このエクステンションは、たとえば実行ファイルや作業ディレクトリなどへのパスの指定を行うため、他のアプリケーションエクステンションですでに定義されている多くのエレメントを再使用する。またこれは、プロセス数、１プロセスあたりのスレッド数、１ホストあたりに実行されるプロセス数を指定する際にも役割を果たす。アプリケーションが必要とする並列環境の種類、たとえば実行に必要な MPI の種類なども識別できる。

2.5 JSDL パラメタスイープエクステンション（ワーキンググループによる草案）
パラメタスイープエクステンションは、JSDL 内の一つあるいは複数のエレメント（パラメタ）の値がどのように変えられて新たな JSDL が作られるのかを定義するものである。したがってこのエクステンションでは、ベースとなる JSDL ドキュメントとスイープ定義とからなる一つのドキュメントで複数のジョブを定義する。

スイープ定義は、一つあるいは複数のパラメタを同時に変えることができ、各パラメタに対して数値配列もサポートしている。ループを定義でき、スイープループのネスティングもサポートしている。したがって非常に多くのジョブを一つのドキュメントにエンコードできる。

この仕様は JSDL のワーキンググループにより現在も検討中であり、2008 年夏には最終版ができると見られている。

2.6 ベーシック実行サービス（BES）（GFD118）

BES の仕様は UNIX や Windows のプロセスのような計算エンティティや並列プログラムを作成、監視、制御するためのインタフェースを定義する。これらはアクティビティ（作業）とよばれるものである。クライアントはアクティビティを JSDL を使って定義する。BES の実装は、受け付けた各アクティビティを実行する。BES リソースが表すものは、一台のコンピュータであったり、ロードシェアリングファシリティ（LSF）、Sun グリッドエンジン（SGE）、ポータブルバッチシステム（PBS）、Windows HPC サーバ 2008（HPCS2008）、Condor などのリソースマネジャで管理されるクラスタであったり、あるいは別の BES 実装であったりする。

2.7 HPCP アプリケーションエクステンション（GFD111）

HPC プロファイルアプリケーションエクステンションの仕様は、HPC リソース上で実行するアプリケーションの定義をサポートするため、JSDL のアプリケーションエレメントに追加内容を記述するものである。とくにこのエクステンションにより実行タスクは次のようなことができるようになる。実行タスクに名前を付ける。実行ファイルや特定のワーキングディレクトリへのパス、実行ファイルへ送る必要のある引数、実行ファイルに必要な環境変数などを実行タスクが決めることができる。標準の入出力ファイルやエラーファイルのパスを実行タスクが指定する。実行タスクを投入したユーザのユーザアイデンティティと異なるユーザアイデンティティを、実行タスクが決めることができ、その下で動作する。

2.8 HPC ベーシックプロファイル（HPCBP）（GFD114）

HPCBP は BES や JSDL 上のプロファイルであり、HPC アプリケーションをクラスタのリソースに投入する際の使用を想定した最低限の機能をサポートするため、HPCP アプリケーションエクステンションを使う。その動作にはデータのステージング、デリゲーション、アプリケーションのデプロイメントなどは含まれない。本文書が書かれた時点でセキュリティモデルに合意は得られていないため、このプロファイルには WS-I BSP（WS-I ベーシックセキュリティプロファイル）のユーザネーム／トークンや X.509 トークンのクレデンシャルプロファイルが認証のために含まれている。このプロファイルは 2006 年の夏から初秋にかけて書かれたが、世界中から集まった 10 以上の異なる実装を使って、Tampa で開かれた SC'06 にて相互運用性のデモンストレーションが行われた。

2.9 HPC ベーシックプロファイルへのファイルステージングエクステンション（GFD135）

JSDL の仕様は、リモートソースやターゲットデータをローカルコピーにマップさせる「データステージング」エレメントをサポートしている。ソースデータの URI が指定されていれば、そのデータは実行タスクがはじまる前にリモートソースからローカルなコピー先へ

コピーされる。ターゲットデータの URI が指定されていれば、そのデータはタスクが実行された後でローカルソースからリモートのコピー先へコピーされる。ユーザは、データがすでにマシン上にローカルに存在していたり、あるいは作業が終了してすでにデータが削除されていたりした場合の動作について定義することができる。

HPCBP ファイルステージングエクステンションは JSDL モデルを用い、共通のファイル移動プロトコル（http(s)、ftp(s)、scp）のための働きを定義する。そしてセキュアなリソースへのアクセスを可能にするために、それぞれのデータのアクションに対するクレデンシャルをどのように与えるのかを、このエクステンションが規定する。これらのクレデンシャルはデータソースやデータシンクによって異なり、また、計算リソースにアクセスするために使用するクレデンシャルとも異なる。仕様に準拠した実装では、WS セキュリティのユーザ名トークンエレメントや X.509 認証トークンエレメントのコード化のうち一つをサポートしていなければならない。

タスクが実行段階の一部としてデータをステージイン・ステージアウトしているかどうかを反映するために、この実装により HPCBP サービスの中で使用する状態モデルを拡張することができる。

HPCBP ファイルステージングエクステンションは一部の主要なプロトコルしか定義しないが、データの移動や名前付けのための GridFTP、DMI(データ移動インタフェース)、e-mail、RNS（リソースネームスペース管理サービス）などの付加的メカニズムをこの実装によりサポートできる。

2.10 ByteIO （GFD87）

ByteIO はバイトシーケンスに対して POSIX 的な読み書き操作を提供するものである。そのインタフェースには 2 種類ある。Random ByteIO インタフェースでは、オフセット、バイト数、データバッファが読み書き操作に送られる。しかし Streamable ByteIO インタフェースでは、読み書き操作はオフセットを受け取らない。

2.11 GLUE （公式発言において）

GLUE2.0 の仕様は、クラスタのリソースを記述するためのモデル、あるいはさらに一般的に、異なるコミュニティが決まったサービスインタフェースを通じて共有するリソースを記述するための明確なモデルを提供する。

2.12 分散リソース管理アプリケーション API （DRMAA）(GFD22)

分散リソース管理アプリケーション API (DRMAA)は、2004 年以降 OGF 標準となってき

た。これは、ジョブの投入や監視関連の API を、ローカルな分散リソース管理システムに提供するものである。詳しくは www.drmaa.org を参照のこと。

2.13 リソースネームスペース管理サービス（RNS）（GFD101）

EPR はアプリケーションの制御に便利ではあるが、すぐに数百文字を超えてしまい、人が使うには不便である。さらに EPR のネームスペースは通常、EPR 間の関係を示していない。これらの欠点を補い、グリッドをより人が使いやすい形にするため、RNS（OGF：リソースネームスペース管理サービス）は、文字列のパスを EPR にマップする階層ディレクトリ構造を提供する。この構造は、Unix のディレクトリが文字列パスを inode にマップするように、文字列パスを EPR にマップする。たとえば、/biology/databases/Sequences/pir21.a という RNS パスを持っているとしよう。このような人が読めるパスを WS アドレッシング EPR にマップするのに RNS が使われるのである。EPR はリソースに直接アクセスするときに使うことができる。

3 シナリオ

この節では、ISV のための主要な HTC シナリオや HPC シナリオを詳しく紹介する。その際、相互運用可能なインフラストラクチャを構築する上で OGF 仕様をどのように用いるのかを説明する。シナリオは以下の要件を満たす。

・実行中のアプリケーションがクライアントワークステーション上のサービスにアクセスできるよう、クライアントワークステーションは計算クラスタがアクセスできる IP アドレスを持つ。

・ジョブの計算ライフサイクル（すなわちジョブの投入、待機、実行）を通じてクライアントワークステーションを使用できる。

それぞれのシナリオに対し、以下の事項を説明する。

・環境：シナリオ内のアプリケーションがどのようにソフトウエアと通信するか。

・前提条件：システムにインストールする必要のあるソフトウエアコンポーネントやソフトウエアパッケージ。

・対話：システムのコンポーネント間の秩序ある対話

・利点：エンドユーザおよびシステム管理者にとっての、シナリオが持つ利点。

3.1 独自のコマンドラインインタフェースを使ったジョブの直接投入

3.1.1 環境

このシナリオでは、サブミッションホストと計算クラスタの間に共有ファイルシステムがあることを想定している。これはスケジューリングソフトウエアの要件になることが多く、多くのエンタプライズデプロイメントにおいて普通のことである。多くのアプリケーショ

ンが、特定のジョブ管理ソフトウエアにカスタマイズできるシェルスクリプトを起動する。たとえば

qsub job_script

などである。

3.1.2 前提条件

サブミッションホストやクラスタ上のオペレーティングシステムに加え、以下のソフトウエアをインストールし、設定する必要がある。

・サブミッションホストおよびクラスタ上の分散リソース管理ソフトウエア
・サブミッションホストおよび計算クラスタ上のアプリケーション

3.1.3 対話

1. 計算クラスタ上で解く必要のある問題を立てるため、ローカルユーザとクライアントアプリケーションとで対話を行う。
2. アプリケーションがリモートクラスタ上で実行されるときに必要とする設定やデータファイルの特定と作成
3. （プログラマチック API を通じて、あるいはコマンドラインツールを直接起動することによって）DRM 固有のインタフェースを使用したジョブの投入、監視、管理
4. アプリケーションあるいはユーザによる出力ファイルの表示または解析

3.1.4 利点

このシナリオは、エンタプライズに完全に含まれる形での ISV のデプロイメントに対し、実行可能なデプロイメントや現実的なシナリオを提示するものである。ローカルのファイルにアクセスできるようにするため、このシナリオが進行する間は、クライアントはネットワークに接続している必要がある。

3.2 標準化 API を使用したジョブの直接投入

3.2.1 環境

このシナリオでは、サブミッションホストと計算クラスタの間に共有ファイルシステムがあることを想定している。これは企業内のジョブスケジューラのデプロイメントに共通する特徴であるが、選んだジョブスケジューラがこれを要求することはほとんどない。さらに、ISV のアプリケーションの中で使われる DRMAA（分散リソース管理アプリケーション API）オペレーションをローカルにインストールされたスケジューラにブリッジするために、スケジューリングベンダーが提供するライブラリが必要となる。

アプリケーション
DRMAA インタフェース
DRMAA プラグイン
スケジューラ

3.2.2 前提条件

サブミッションホストやクラスタ上のオペレーティングシステムに加え、以下のソフトウエアをインストールし、設定する必要がある。

・サブミッションホストおよびクラスタ上の分散リソース管理ソフトウエア（ジョブスケジューラ）
・サブミッションホストおよび計算クラスタ上のアプリケーション
・ローカルにインストールされた DRM ソフトウエアのために DRMAA 仕様を実装したダイナミックライブラリをクライアントにインストールする必要がある。

3.2.3 対話

1. 計算クラスタ上で解く必要のある問題を立てるため、ローカルユーザとクライアントアプリケーションとで対話を行う。
2. アプリケーションがリモートクラスタ上で実行されるときに必要とする設定やデータファイルの特定と作成
3. DRMAA のプログラマチックインタフェースを使用したジョブの投入、監視、管理
4. アプリケーションあるいはユーザによる出力ファイルの表示または解析

3.2.4 利点

このシナリオは、エンタプライズに完全に含まれる形での ISV のデプロイメントに対し、実行可能なデプロイメントや現実的なシナリオを提示するものである。このシナリオは、ISV のためにプログラマチックインタフェースを与える。このインタフェースは使われているスケジューリングソフトウエアとは独立なものであるが、これはすなわち、アプリケーションが実装されると、そのアプリケーションは DRMAA 仕様に準拠しているいかなるスケジューリングソフトウエアとも動作するということを意味する。ISV 開発者が SAGA（Simple API for Grid Application）のようなよりハイレベルの API を使うこともあり、こうした API がスケジューラインフラストラクチャにアクセスするために DRMAA（あるいは他の API）を内部的に使用する場合がある。このシナリオは、プラットフォームに固有な形でジョブスケジューラとの統合を行うという作業を、アプリケーション ISV からスケジューリング ISV に移すことになる。アプリケーション ISV は DRMAA インタフェースに関する仕事を行い、その一方で、スケジューラ ISV は DRMAA インタフェースをネイテ

ィブのスケジューリングインタフェースにリンクするプラグインを提供する。

3.3 ウェブサービスを通じたジョブの直接投入

3.3.1 環境

このシナリオでは、サブミッションホストと計算クラスタの間に共有ファイルシステムがあることを想定している。アプリケーション ISV は、計算クラスタへのアクセスを提供するウェブサービスエンドポイントと通信を行うため、ウェブサービスのプロトコルとインタフェースを使う。この働きを定義するために使われる主な仕様としては、HPC ベーシックプロファイル仕様（およびそれに依存した仕様、たとえば BES、JSDL、HPCP アプリケーション仕様）がある。このウェブサービスには、ウェブサービスインタフェースを直接使うアプリケーションがアクセスしたり、あるいは中間ライブラリを使ってアクセスしたりできる。

3.3.2 前提条件

サブミッションホストやクラスタ上のオペレーティングシステムに加え、以下のソフトウエアをインストールし、設定する必要がある。

・サブミッションホストおよび計算クラスタ上のアプリケーション
・クライアントにインストールされたウェブサービスクライアント環境あるいはサブミッションホスト上のアプリケーションに統合されたウェブサービスクライアント環境
・計算クラスタ内でジョブを投入、監視、管理することができるよう、企業ネットワークにインストールされた環境やサービスをホスティングするウェブサービス
・クラスタ上のウェブサービスはクライアントからアクセスできる必要がある。

3.3.3 対話

1. 計算クラスタ上で解く必要のある問題を立てるため、ローカルユーザとクライアントアプリケーションとで対話を行う。
2. アプリケーションがリモートリソース上で実行されるときに必要とする設定やデータファイルの特定と作成
3. 以下を用いた JSDL ドキュメントの作成
a. シングルプロセッサジョブの場合、HPC プロファイルアプリケーション
b. マルチプロセッサジョブの場合、JSDL-SPMD
c. パラメタスイープジョブの場合、JSDL-パラメタスイープ仕様
4. クラスタの HPCBP エンドポイントへの JDSL ドキュメントのサブミッション
5. アプリケーションの実行
6. アプリケーションあるいはユーザによる出力ファイルの表示または解析

3.3.4 利点

このシナリオは、クライアント上の依存ソフトウエアを最低限必要とする実行可能なデプロイメントを表わしている。クライアント側のウェブサービススタックはアプリケーションの中に完全に含めてしまうことができ、DRM 専用ソフトウエアとそれをとりまくインタフェースやラッパーをセットで導入する必要はない。ウェブサービスは、企業ネットワークで必要であり、クライアントが DRM の中でジョブを投入、監視、管理するためにアクセスするものである。結果として、アプリケーション ISV のために環境とのよりプレディクタブルな対話を提供できる、これまでよりもシンプルなクライアントとなる。

3.4 ファイルステージングを用いるウェブサービスを通じたジョブの直接投入

3.4.1. 環境

このシナリオは、クライアントワークステーションと計算クラスタの間に共有ファイルシステムがあることを想定していない。アプリケーション ISV は、計算クラスタへのアクセスを提供するウェブサービスエンドポイントと通信を行うため、ウェブサービスのプロトコルとインタフェースを使う。この働きを定義するために使われる主な仕様としては、HPC ベーシックプロファイル仕様（およびそれに依存した仕様、たとえば BES、JSDL、HPCP アプリケーション仕様）と HPCBP ファイルステージングエクステンションがある。このウェブサービスには、ウェブサービスインタフェースを直接使うアプリケーションがアクセスしたり、あるいは中間ライブラリを使ってアクセスしたりできる。さらに、アプリケーションが実行時に必要とするいずれのファイルも、外部ストレージサーバを通じてクラスタに明示的に移される。

3.4.2 前提条件

クライアントワークステーションやクラスタ上のオペレーティングシステムに加え、以下のソフトウエアをインストールし、設定する必要がある。

・クライアントワークステーションおよび計算クラスタ上のアプリケーション

・クライアントにインストールされたウェブサービスクライアント環境あるいはクライアントワークステーション上のアプリケーションに統合されたウェブサービスクライアント環境

・計算クラスタ内でジョブを投入、監視、管理することができるよう、企業ネットワークにインストールされた環境やサービスをホスティングするウェブサービス

・クライアントワークステーションと計算クラスタの両方でファイル移動プロトコルクライアントがサポートしているファイル移動プロトコルサービスのうち、少なくとも一つをサポートする外部ストレージサーバ

3.4.3 対話

1. 計算クラスタ上で解く必要のある問題を立てるため、ローカルユーザとクライアントアプリケーションとで対話を行う。
2. アプリケーションがリモートリソース上で実行されるときに必要とする設定やデータファイルの特定と作成
3. 以下を用いた JSDL ドキュメントの作成
a. シングルプロセッサジョブの場合、HPC プロファイルアプリケーション
b. マルチプロセッサジョブの場合、JSDL-SPMD
c. パラメタスイープジョブの場合、JSDL-パラメタスイープ仕様
さらに、サポートされているプロトコルのうちの一つを使い、外部ストレージサーバから計算クラスタに入力および（または）出力の依存ファイルを移す HPCBP ファイルステージングエクステンションも使用する。
4. クライアントの要求があった場合、クライアントと外部ストレージサーバがサポートするプロトコルを使って指定された外部ストレージサーバへファイルをアップロードする。
5. クラスタの HPCBP エンドポイントへの JDSL ドキュメントのサブミッション

クライアントワークステーションはこの時点でネットワークから切り離してもよい。そしてクラスタ上で非同期的に次のステップが始まる。次のステップは、計算クラスタ上のミドルウエアによってローカルファイルシステムを通じて進められる。
6. 外部ストレージサーバからクラスタへのファイルのダウンロード
7. アプリケーションの実行
8. 指定された外部ストレージサーバへのあらゆる出力ファイルのアップロード

クライアントワークステーションは次のいずれかを行う。
a) ネットワークに接続したままで、作業が完了したということをポーリングを通じて知る。
b) ネットワークに再接続し、ワークステーションが送った作業が完了したことを知る。

クライアントワークステーション上では以下の作業が引き続き実行される。
9. 指定した外部ストレージサーバからクライアントワークステーションへ出力ファイルをとってくる。
10. ジョブが完了したことをユーザに（ローカルに）通知。

3.4.4 利点

このシナリオでは、ジョブが投入されるとクライアントワークステーションをネットワークから切り離し、ジョブが完了すると出力ファイルをダウンロードするためにネットワー

クに再接続するということができる。DRM 専用ソフトウエアをクライアントワークステーションに導入する必要はない。クライアント側のウェブサービススタックはアプリケーションの中に完全に含めてしまうことができ、DRM 専用ソフトウエアとそれをとりまくインタフェースやラッパーをセットで導入する必要はない。ウェブサービスは、企業ネットワークで必要であり、クライアントが DRM の中でジョブを投入、監視、管理するためにアクセスするものである。

3.5 ランタイム対話を用いるウェブサービスを通じたジョブの直接投入

3.5.1 環境

このシナリオは、クライアントワークステーションと計算クラスタの間に共有ファイルシステムがあることを想定していない。アプリケーション ISV は、計算クラスタへのアクセスを提供するウェブサービスエンドポイントと通信を行うため、ウェブサービスのプロトコルとインタフェースを使う。この働きを定義するために使われる主な仕様としては、HPC ベーシックプロファイル仕様（およびそれに依存した仕様、たとえば BES、JSDL、HPCP アプリケーション仕様）がある。このウェブサービスには、ウェブサービスインタフェースを直接使うアプリケーションがアクセスしたり、あるいは中間ライブラリを使ってアクセスしたりできる。クライアントワークステーションと計算クラスタ上で実行されているアプリケーションとの双方向対話が、計算クラスタのヘッドノード上で動いている中間 ByteIO サービスを通じて行われる。

3.5.2 前提条件

クライアントワークステーションやクラスタ上のオペレーティングシステムに加え、以下のソフトウエアをインストールし、設定する必要がある。

・クライアントワークステーションおよび計算クラスタ上のアプリケーション
・クライアントにインストールされたウェブサービスクライアント環境あるいはクライアントワークステーション上のアプリケーションに統合されたウェブサービスクライアント環境
・計算クラスタ内でジョブを投入、監視、管理することができるよう、企業ネットワークにインストールされた環境やサービスをホスティングするウェブサービス
・クライアントワークステーションと計算クラスタ内の計算ノードの両方にアクセスできる ByteIO

3.5.3 対話

1. クラスタリソース上で解く必要のある問題を立てるため、ローカルユーザとクライアントアプリケーションとで対話を行う。

2. アプリケーションがリモートリソース上で実行されるときに必要とする設定やデータファイルの特定と作成
3. クラスタのヘッドノード上の ByteIO エンドポイントの初期化
4. ByteIO エンドポイントを用いた JSDL ドキュメントの作成
a. シングルプロセッサジョブの場合、HPC プロファイルアプリケーション
b. マルチプロセッサジョブの場合、JSDL-SPMD
c. パラメタスイープジョブの場合、JSDL-パラメタスイープ仕様
5. クラスタの HPCBP エンドポイントへの JDSL ドキュメントのサブミッション

計算クラスタ上の作業とクライアントワークステーション上の作業は平行して進む。クライアントと実行中のアプリケーションとの間に双方向チャネルができ、これによりクライアントワークステーションがアプリケーションを「操縦」し、アプリケーションがクライアントに中間結果を報告することができる。

計算クラスタ側	クライアントワークステーション側
6. アプリケーションの初期化	
7. 中間結果をヘッドノードの ByteIO エンドポイントに送る。	ByteIO エンドポイントから中間結果を取得し、それに基づいて作業を行う。
8. アプリケーションの完了	アプリケーションが終了するまでその作業をポーリングする。

クライアントワークステーション上では以下の作業が引き続き実行される。
9. ジョブが完了したことをユーザに（ローカルに）通知。

3.5.4 利点

このシナリオでは、クライアントワークステーションが実行中のアプリケーションから中間結果を取得することと、そのアプリケーションに制御コマンドを送ることの両方が可能である。共有ファイルシステムは必要とせず、クライアントワークステーションと実行中のアプリケーションの両方から見える ByteIO サービスだけが必要となる。DRM 専用ソフトウエアをクライアントワークステーションに導入する必要はない。クライアント側のウェブサービススタックはアプリケーションの中に完全に含めてしまうことができ、DRM 専用ソフトウエアとそれをとりまくインタフェースやラッパーをセットで導入する必要はない。ウェブサービスは、企業ネットワークで必要であり、クライアントが DRM の中でジョブを投入、監視、管理するためにアクセスするものである。

クライアントワークステーションがクラスタにより直接アドレス可能である必要はなく、共有ファイルシステムも必要としない。アプリケーションは、中間出力を ByteIO エンドポ

イントに送るために修正を必要とする。クライアントは NAT 機能付きのファイアウォールのもとで作業を行うことができるが、実行中のアプリケーションはそれでもクライアントに結果を返すことができる。

4 実装アーキテクチャ

4.1 クライアントワークステーション

クライアントマシンの管理者は、アプリケーションソフトウエアのクライアントコンポーネントと、クライアントからシステム内の他のコンポーネントへの対話に必要なミドルウエアソフトウエアのクライアントコンポーネントをインストールする必要がある。ミドルウエアのコンポーネントには以下のものが含まれる場合がある。

・JSDL&HPC プロファイルアプリケーションエクステンション：リモートに実行するジョブ（実行タスクやデータ移動）を定義するスキーマ

・HPC ベーシックプロファイル：リモートのクラスタリソースと通信するウェブサービスクライアントプロトコル

・データムーブメント（データ移動）：ファイルやデータを、クライアントへ、あるいはクライアントから外部ストレージサービスへ移動するときに使用するクライアントプロトコル

・ByteIO：ヘッドノード上の ByteIO サービスへ情報を送る、あるいは ByteIO から情報を受け取るのに必要なウェブサービスクライアントプロトコル

さらに、クライアントがクラスタリソースに対して本人確認するために使うローカルなクレデンシャルを、クライアントワークステーション環境が確認する必要がある。クラスタのヘッドノードでクライアントを確認するために使用できるローカルなクレデンシャルがない場合は、代わりとなるクレデンシャルが必要になる。クレデンシャルが確認されると WS セキュリティプロトコルを通じてリモートサービスに受け渡される。

4.2 外部ストレージサーバ

前述のシナリオで使う主なファイルなどのデータを交換するため、外部ストレージサーバは、クライアントワークステーションと計算ノードの両方にアクセス可能な場所を提供する。外部ストレージサーバは、クライアントワークステーション環境とアプリケーション（計算ノード）環境の両方がサポートしているデータ移動サービスのうち、少なくとも一つをサポートしている必要がある。

4.3 計算クラスタのヘッドノード

クラスタの管理者は、「ミドルウエア」ソフトウエアのサーバ側のコンポーネントについて、一回限りのインストールと設定を行う必要がある。このソフトウエアは、アプリケーショ

ン、オペレーティングシステム、クラスタ管理ソフトウエアを結びつける働きをするものである。（ローカルリソースマネジャが必要とするものに加えて）ヘッドノード上のサービスには以下のものが含まれる場合がある。
・HPC ベーシックプロファイルサービス：クラスタでジョブを投入、管理するウェブサービス。これにはこのサービスに投入される JSDL ドキュメントの解析も含まれる。
・ByteIO サービス：アプリケーションの実行中にデータを交換するため、クライアントワークステーションとアプリケーションに対して「ランデブー」ポイントを与えるウェブサービス。これは、アプリケーションがクライアントワークステーションに中間結果を送る場合、そして計算ノードがクライアントワークステーションに直接アクセスできない場合にのみ必要となる。

4.4 計算クラスタの計算ノード上で実行されているアプリケーション
アプリケーションは以下のものをサポートする必要がある。
・ByteIO：アプリケーションの実行中にアプリケーションと情報のやりとりを行うのに必要なウェブサービスクライアントプロトコル。これは、アプリケーションがクライアントワークステーションに中間結果を送る場合にのみ必要である。実行中のアプリケーションがクライアントワークステーションに直接アクセスできない場合は、ランデブーポイントとしてヘッドノード上の ByteIO サービスを使うことができる。
・データ移動：外部ストレージサーバにデータを移動する、あるいは外部ストレージサーバからデータを移動するために必要なクライアントプロトコル。これはアプリケーションの中に埋め込むこともでき、あるいは HTTP、HTTPS、FTP、GridFTP など、クラスタの実行インフラストラクチャの一部として実装することもできる。

4.5 導入したソフトウエア環境の検証
これらのシナリオのいずれかを試みる前に、クラスタとクライアントマシンの両方で行うべきいくつかの検証ステップを以下に示す。

・（「ミドルウエア」は使用せずに）ヘッドノードあるいはサブミッションホストからクラスタ上でアプリケーションを走らせる。
〇シングルプロセッサ上で
〇マルチプロセッサ上で並列（MPI）ジョブの一部として
・ヘッドノードから外部ストレージサーバへファイルをアップロードする。
・外部ストレージサーバからヘッドノードへファイルをダウンロードする。
・計算ノードから外部ストレージサーバへファイルをアップロードする。
〇計算ノードから外部ネットワークが見えない場合は、クラスタはこの機能をサポートしないことがある。

・外部ストレージサーバから計算ノードへファイルをダウンロードする。
〇計算ノードから外部ネットワークが見えない場合は、クラスタはこの機能をサポートしないことがある。
・クライアントマシンから外部ストレージサーバへファイルをアップロードする。
・外部ストレージサーバからクライアントマシンへファイルをダウンロードする。
・クライアントマシンからクラスタへのネットワーク接続がある。
・クライアントマシンからクラスタのヘッドノードへファイルをアップロードする。
・クラスタのヘッドノードからクライアントマシンへファイルをダウンロードする。
・クライアントマシン上でユーザが与えるクレデンシャルが、ヘッドノード上のミドルウエアへアクセスできる。
・クライアントマシン上でユーザが与えるクレデンシャルが、ヘッドノード上のサービスを通じてクラスタへアクセスし、ジョブを投入できる。

これらのテストをインストール時と使用時に利用することで、クライアントが現状のネットワーク環境に適応するよう自己設定する手段や、不具合を解決する手段が得られることになる。

5 ISV への助言
本文書は、分散コンピューティング環境に共通の使用シナリオに対し、確立済みのウェブサービス標準や新しいウェブサービス標準を使って系統的なアプローチを提供するものである。（プラットフォームプロバイダから）このインフラストラクチャが得られれば、ISV は、アプリケーションを通じて提供するドメイン固有の機能性に集中することができる。複雑なファイアウォールのあるネットワークでデータを動かすという複雑さにはとらわれずに済むのである。

これらのウェブサービス標準の中でクライアントに関係する部分を実装するのは、オープンソースツールキットあるいは商用的なサポートのあるサンプルコードや枠組み等があれば、比較的単純である。

以下の表に書かれた計画や製品は、団体のサポートや商用的サポートにより、いくつかの異なるオペレーティングシステム（たとえば Java と C Apache Axis、Windows Communication Framework、gSOAP など）にまたがって、いろいろなオープンソースやクローズドソースのツールキットを使用する。これらの多様な技術を用いて OGF 団体が得た相互運用性に関する経験は、すでに文書にまとめて標準の策定の助けとしたり、製品や計画が使用するソフトウエアに統合されたりしてきた。

以下の表は本文書で議論した仕様の(2008 年 5 月現在の)サポート状況を示すものである。これらの計画はすべて、必要であれば WS アドレッシングを使用し、認証レベルセキュリティ、トランスポートレベルセキュリティ、メッセージのレベルセキュリティのために WS セキュリティのエレメントを使用する。SPMD やパラメタスイープアプリケーションのための JSDL エクステンションは、まだ採用の途についたばかりである。HPC ベーシックプロファイルは HPC プロファイルアプリケーションエクステンションのサポートを含む。DRMAA 仕様のサポート状況は、http://www.drmaa.org に記録がある。

仕様 計画あるいは製品	OGSA-BSP 2.0	JSDL	OGSA-BES	HPCBP	ファイルステージング	OGSA-ByteIO	RNS
Globus	いいえ	はい	はい	はい	いいえ	いいえ	いいえ
UNICORE 6	いいえ	はい	はい	はい	いいえ	はい	？
USMT (富士通)	予定あり		はい	はい	はい	いいえ	はい	？
HPCS 2008 (Microsoft)	いいえ	はい	はい	はい	はい	いいえ	いいえ
Genesis II	予定あり		はい	はい	はい	はい	はい	はい
GridSAM (OMII-UK)	いいえ	はい	はい	はい	はい	いいえ	いいえ
Crown	いいえ	はい	はい	はい	いいえ	いいえ	いいえ
BES++ (Platform)	いいえ	はい	はい	はい	はい	いいえ	いいえ
NAREGI	いいえ	はい*	いいえ	いいえ	いいえ	いいえ	？
Gfarm	いいえ	いいえ	いいえ	いいえ	いいえ	予定あり		いいえ
gLite	いいえ	プロトタイプ	プロトタイプ	プロトタイプ	いいえ	プロトタイプ	？
ARC1 (NorduGrid)	予定なし		はい	はい	はい	いいえ	はい	いいえ

図 3：いくつかの計画および製品における仕様のサポート状況
*JSDL SPMD をサポート
+JSDL パラメタスイープをサポート

6 プラットフォームプロバイダへの助言

本文脈におけるプラットフォームプロバイダとは、ISV のアプリケーションがクラスタリソース上で作業を開始するために使用するミドルウエアのサプライヤを指す。プラットフォームプロバイダは、クラスタのジョブスケジューリングソフトウエアと統合したオープンソースプラットフォームプロバイダである場合や、ミドルウエアの仕様をサポートするジョブスケジューリングソフトウエアのプロバイダである場合がある。

これらの仕様をサポートすれば、系統的かつ相互運用可能な形で標準化された方法により、分散計算インフラストラクチャの ISV のアプリケーションがミドルウエアを使用できるようになる。しかしこれらの使用をサポートしなければ、このような方法での使用は多くのネットワーク環境で現在のところ不可能である。

7 未解決の問題

7.1 直接ジョブを投入する場合のリソースの選び方

第 3 節に示したシナリオでは、どの HPCBP リソースを使用するのかがあらかじめわかっていた。企業は多くの異なる HPCBP リソースを持つことがたびたびあり、そのそれぞれが別の物理的計算クラスタを表している。各 HPCBP リソースは独自の WS アドレッシングのエンドポイントリファレンス（EPR）を持っている。この EPR は、第 3.3 節、3.4 節、3.5 節に説明したように、ウェブサービス対話の相手として使用する。問題は、クライアントが使用する HPCBP リソースを選ぶことにあり、それにはいくつかの方法がある。以下に 4 つのアプローチを紹介する。その 4 つとは、情報サービスのクエリ、RNS パス名の使用、メタスケジューリング HPCBP リソースの使用、OGSA-RSS サービスの使用である。

7.1.1 情報サービスのクエリ

情報サービスは、たとえば OperatingSystem=Windows、PhysicalMemory>2GB など、リソースを記述する言葉を使ってクエリする。これらの要件に合うリソースが XML ドキュメントに戻される。GLUE 仕様は、分散計算性能やストレージ性能を記述するための豊富なスキーマの一例である。現在のところ HPCBP は（背景にあるベーシック実行サービスを通じて)、エンドポイントに含まれるリソースを記述するための限られた基本スキーマをサポートしている。各 XML ドキュメントには、関係のある HPCBP とクエリを満たす HPCBP リソースの EPR とが含まれている。クライアントが HPCBP のエンドポイントの一つを選択し、シナリオが先述したように進行する。

このアプローチでは、ユーザが指定したクエリに合うリソースが、どのリソースを使用するかを判断したクライアントに返される。

7.1.2 RNS パスの横断

リソースネームスペースサービス（RNS）が供給するディレクトリ構造にさまざまな HPCBP リソースが置かれる。RNS は階層ネームスペース（すなわちディレクトリ構造）を与え、これがパス名を EPR にマップする。たとえば RNS のディレクトリ“/clusters/USNationalCenters”には”TACC”、”IU”、”NCSA”などのエントリが置かれ、それぞれ TACC、IU、NCSA のクラスタを表す HPCBP リソースを指す。ユーザは、

クライアントソフトウエアを用いて RNS ネームスペースを検索し、“/clusters/USNationalCenters/TACC”などのように、使いたいクラスタを選ぶことができる。HPCBP エンドポイントが選ばれると、シナリオが先述したように進行する。

このアプローチでは、ユーザは階層的に並べられたリソースを検索し、使用するリソースを選ぶことができる。

7.1.3 メタスケジューリング HPCBP リソース

このシナリオでは、ある HPCBP リソースが、他の多くの HPCBP リソースをカプセル化するプロキシとして働く。この“メタスケジューリング”HPCBP リソースは、クライアントの要求を受け、準拠した HPCBP リソースを何らかの内部ポリシにしたがって選択し、その選んだ HPCBP リソースにジョブをデリゲートする。この作業は、クライアントからは、メタスケジューリング HPCBP リソースが自らジョブを行っているように見える。セキュリティデリゲーションが問題になることがあるが、HPC-BP ファイルステージングエクステンションが使われていればデータアクセスクレデンシャルを JSDL に入れられることに注意しよう。

このアプローチでは、どのリソースにジョブを送るかということについて内部的に決めるリソースに対して、ユーザがジョブを投入する。

7.1.4 リソース選択サービス

OGSA アーキテクチャ文書の OGSA 実行管理サービスの節では、実行アーキテクチャを説明している。これには「ジョブマネージャ」、実行サービス（HPCBP など）、情報サービス、リソース選択サービス（RSS）が含まれる。リソース選択サービスは、JSDL ドキュメントにあるようなジョブ記述を受ける役割があり、実行サービスの候補となるものを決めたり、（「良い」ものだけを選び出す何らかの最適化機能により）ジョブを実行するのに最良のリソースを選んだりする。OGSA-RSS ワーキンググループが、提案されたインタフェースの開発を行ってきた。

このアプローチでは、ユーザがリソースの要件を決めることができ、そして RSS が一つあるいは複数の選ばれた HPCBP リソースの一覧を示す。

7.1.5 利点

これらのアプローチの利点は、クライアントが特定のリソースに縛られることがないということである。リソースが必要になるたびに、その時点でユーザが使うことのできるリソースの中から適切なリソースが選び出される。クライアントが特定の「既知の」計算リソ

ースについて知っておく必要はないが、その代わりに、使用する計算クラスタの詳細を含んだ情報の「既知の」ソースについては知っておく必要がある。

7.2 ウェブサービスインフラストラクチャとの対話

アプリケーション開発者がウェブサービスインフラストラクチャとどのように対話するかも未解決の問題である。専用のサービススタックを用いたウェブサービスインタフェースを直接使って生成したプログラマチック API か、あるいはどのサービススタックでも実装可能なポータブルインタフェースを通じて生成したプログラマチック API のどちらかを通じて対話するのだろうか。あるいは、ウェブサービスとの対話をカプセル化するソフトウエアサプライヤが供給する一連のコマンドラインツールを、アプリケーションから起動するということもできる。

7.3 既存のセキュリティインフラストラクチャとの対話

デプロイするウェブサービスインフラストラクチャは、それがどのようなものであっても、既存のサービスインフラストラクチャと統合しなければならない。特定のアプリケーションやウェブサービスをサポートするために新しいセキュリティインフラストラクチャを用いるようなことは行わない。したがって問題は、デプロイした環境において現在見ることのできる「ネイティブの」セキュリティメカニズムは何か、そしてそのメカニズムはウェブサービスの枠組みからどのように透過的に見つけることができるか、ということである。さらに、これらのセキュリティトークンは、計算クラスタや外部ストレージサーバで認証検査や許可検査を行う上でどのように使用されるのか、そしてファイル転送プロトコル（ftp、scp、http など）にどのように統合することができるか、ということも問題になる。

8 セキュリティ事項

本文書に記載したシナリオには、モバイルクライアント、貴重な計算リソース、潜在的商業価値の高いシミュレーションデータなど、重要なセキュリティ上の課題がある。さらにこれらのシナリオを展開するために配備する必要のあるミドルウエアは、既存のセキュリティインフラストラクチャ（およびソフトウエアインフラストラクチャ）と一体化されなければならない。

本文書で言及したウエブサービスツールキットの多くは、WS セキュリティ仕様のエレメントを用いている。この仕様の使用は、WS-I BSP（ベーシックセキュリティプロファイル）に基づくより広いウェブサービス団体や、OGSA-BSP2.0 に基づくグリッドコミュニティにより、さらに制限を受ける。本文書の個々の仕様は、それがサポートするセキュリティモデルを定義することに特化してきた。HPC ベーシックプロファイルとファイルステージングエクステンションは、WS セキュリティのユーザネームトークンとパスワードトークンお

よび X.509 証明書トークンのプロファイルを使用する。

9 著者情報

Steven Newhouse
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
steven.newhouse@microsoft.com

Andrew Grimshaw
University of Virginia
Charlottesville, VA
USA
grimshaw@virginia.edu

10 貢献者および謝辞

本仕様への寄与および本仕様に関する議論に対し、Narfi Stefansson および UVA ワークショップや OGSA ワーキンググループの他のメンバーに謝意を表する。

本書の草案を読み、コメントを寄せてくれた方々へも感謝したい。彼らのコメントは有用であり、本文書の読みやすさや正確性を高めることになった。

11 著作権情報

上記の制限付き許諾は永続的なものであり、OGF あるいはその後継団体や譲受団体により破棄されることはない。

本文書とそこに含まれる情報は、「現状」をもとに提供されるものである。オープングリッドフォーラムは、明示あるいは暗示を問わず、あらゆる保証を拒否する。これには、ここに含まれる情報の使用がいかなる権利も侵害しない保証や、特定目的に対する商品性や適合性の暗示保証なども含まれるが、これらにかぎられるわけではない。

12 知的所有権について

本文書に記載された技術の実装や使用に関係すると考えられるすべての知的所有物やその他の権利の有効性や範囲について、あるいはこれらの権利のもとで許諾が得られる範囲について、OGF はいかなる立場も取らない。さらに OGF は、これらの権利を確認する努力を払ってきたと明言することはない。得られる許諾の公表や保証に対する請求権の写し、あるいは知的財産権使用の一般的な許諾や許可を得るために本書の仕様を実装した者あるいは使用した者が行った試みの結果は、OGF 事務局から取得することができる。

OGF はすべての関係者に対し、本書の推奨事項を実施する上で必要となる可能性のある技術に関係する著作権、特許、特許出願、その他の所有権に注意を払うことを求める。情報は OGF 事務局長に寄せていただきたい。（OGF のウェブサイトの連絡先を参照のこと。）