

分類: 情報提供型

グリッドコンピューティング環境 - RG

G. Fox, インディアナ大学、コミュニティグリッド研究室

M. Pierce, インディアナ大学、コミュニティグリッド研究室

D. Gannon, インディアナ大学、CS & PTL

M. Thomas, TACC, テキサス大学、オースティン

2003年2月

グリッドコンピューティング環境の概要

このレポートの位置付け

このレポートは、グリッドシステムへのポータルアクセスに関心を持つグリッドコミュニティへの情報提供を目的としています。配布は無制限です。

著作権表示

Copyright (C) Global Grid Forum (2002). All Rights Reserved.

本書について

このドキュメントは、約 50 件の研究論文から得られた、グリッドコンピューティング環境におけるベストプラクティスを調査したものです。ベストプラクティスは、アーキテクチャの原理、すなわち多層サービスベースモデル、メタデータの役割、ワークフロー、GCEShell を形成するツールや中核機能、アグリゲーションポータル等の観点から説明しています。今後これらの多くが、別のドキュメントでさらに改良されていくことを期待します。

目次

本書について	1
1. はじめに	4
2. GCE システムの全体的な分類	5
3. GCE プロジェクトの概要と機能	9
3.1 GCE システム構築のための技術	9
3.2 広範な問題解決環境	10
3.3 広範な基本 GCEShell ポータル	11
3.4 ワークフロー	12
3.5 データ管理	13
3.6 GCEShell ツール	13
4. GCE コンピューティングモデル	16
4.1 GCE モデルに関する調査	16
4.2 2 層プログラミングモデル	19
5. ポータルサービス	22

5.1GCE ポータル対応のオープングリッドサービスアーキテクチャの実装 24

6. セキュリティ要件 25

著者の連絡先 26

知的所有権表示 27

著作権全文表示 28

参考資料 29

1. はじめに

このドキュメントは、グリッドコンピューティング環境の現在の状況を要約したものです。最近の書籍[37]から抜粋した 15 章[38-52]と、2002 年に機関誌『Concurrency and Computation: Practice and Experience』[54]の特別号として発行された、グローバルグリッドフォーラム(GGF)の GCE (グリッドコンピューティング環境)研究グループ[55]が収集した 29 件の研究論文[1-28]の調査[36, 38] をまとめたものです。グリッドは、概念や実装面で急速な発展をとげ、グリッドシステムについて考える際の「正しい」方法に関して大きな反響や混乱が生じています。グリッドコンピューティング環境 (GCE) は、ひと言でいえば、図 1 に示すようなコアグリッドと呼ばれるコンピューティングシステムのうち、「ユーザーサイド」のコンピューティングシステムを記述するものです。GCE 間には不明瞭な部分が存在し、図では「コア」グリッドと呼んでいます。後者には資源へのアクセスや資源の管理、相互の対話、セキュリティその他の機能が含まれます。こうした「コア」機能は、新しいオープングリッドサービスアーキテクチャ (OGSA) [56](現在も展開中)で説明され、Globus プロジェクト[32]はもっとも有名な「コア」ソフトウェアプロジェクトです。

ここでは、グリッドコンピューティング環境について、ユーザーが「容易」にグリッドの資源やアプリケーションにアクセスできる一連のツールや技術を定義することにします。ユーザーにとっては、多層グリッドアプリケーション開発スタックにユーザーインタフェースを提供する Web ポータルのように見えることが多いかも知れませんが、Grid Shell と同じくらい簡単でもあるのです。Grid Shell では、ユーザーが従来のシェルでファイルシステムや通常のオペレーティングシステムのプロセス空間にアクセスできるのと同じように、グリッドの資源へのアクセスや管理を行えます。このドキュメントで参照している研究論文には、いずれも図 1 と同じような図が描かれていますが、使用する技術(Python に対する Perl の使用など)や言及している機能、プログラム(バックエンドの資源)ではなくユーザーの側面を強調している点で異なります。

上に述べたように、GCE は(少なくとも)次の 2 つの機能を実現します。

「グリッドのユーザー側のプログラミング」-このドキュメントの第 2~4 節で取り上げるテーマです。

ユーザーインタラクションの管理 - アウトプットの表示と Web ページでのユーザー入力。これには単一のポータルページにおける複数のデータソースの集約が含まれます。GCE のこの側面については第 5 節で取り上げます。

図 1. 「コアグリッド」およびグリッドコンピューティング環境を表すアプリケーションの中間層および raw (HPC) リンクコンポーネント

Database	データベース
Database Service	データベースサービス
Compute Service	計算サービス
MPP Service	MPP サービス
Portal	ポータル
HPC or “Native” Linkage	HPC または「ネイティブ」リンケージ
Middle Tier or Proxy Linkage	中間層またはプロキシリンケージ
Grid Computing Environments	グリッドコンピューティング環境
“Core” Grid	「コア」グリッド

2. GCE システムの全体的な分類

グリッドコンピューティング環境は、いくつかの異なる方法で分類できます。簡単な方法としては、使用している技術に基づく分類です。使用する言語や(場合によっては)オブジェクトの処理特性、Java サブレット、Globus ツールキット、GridFTP など特定の技術の使用、その他の実装結果によって、プロジェクトが区分されます。そのうちのいくつかの要素は、パフォーマンスやアーキテクチャにとって重要なものですが、ユーザーにとってはそれほど重要でない場合も少なくありません。たとえば、Java や XML、Web サービスが多用される傾向が見られますが、実際に関心もたれるのは完成するシステムに、カスタマイズや持続力の優秀性、パフォーマンスのような領域への影響が少ない軽快な操作性などの重要な特性がある場合だけに限られます。最新技術を使えば開発も容易になるため、通常 GCE では所定の実装作業だけでより優れた機能を実現できます。プロジェクトにおける技術の差異も大切ですが、この段階では、明示、黙示を問わず、GCE における機能とコンピューティングモデルの差異がより重要と考えられます。

すべての GCE システムは、なんらかのバックエンドのリモート資源(グリッド)が存在するものと仮定し、その機能に対するアクセスの利便性を提供しようとしています。ここに、ある種の「コンピューティ

ング」のためのモデルが必要となります。もっとも単純な例としてはジョブの実行があります。まず重要なことはデータは常に正しくセットアップされていなければならない、次に実行中のジョブのステータスや最終的なアウトプットへのアクセスが可能である必要があります。より複雑な例では、調整されたデータ収集、多くのシミュレーション(既定の時間でリンクまたは相互にフォロー)、視覚化、結果の分析等が必要とされます。またこうした研究の中には、研究者間の緊密な共同作業や、結果やアイデアの共有が必要なものもあります。ここから、同じ問題領域について研究を行う科学チーム同士が共用できる、GCE 共同実験室の概念が生まれます。

1台のコンピュータ上でのある種の一般的なタスク処理として問題をとらえることで、異なる GCE アプローチを想定できます。ここでは以下のような機能について説明します。

1)分散コンピューティングシステムの基本要素、すなわちファイル、計算資源、データ資源、プログラム、アカウントなどの処理。GCE では、バックエンド資源の処理に、Globus や PBS のバッチスケジューラのような環境とのインタフェースを取るのが一般的です。しかし、GCE ではこうした資源を制御するユーザーインタフェースを提供します。インタフェースは、このような方法で構築したツールを反映させるために、簡素にも複雑にもすることができ、さらには階層化することもよくあります。UNIX のリード(分散エクステンションでは Legion [43])に従って、分散コンピューティングの中核機能へのアクセスを提供する基本的な GCEShell を定義することもできます。

たとえば、JXTA [35]も UNIX シェルモデルを使ってグリッドのような機能を構築します。GCEShell はジョブの実行やコンパイル、ファイルシステム間の移動などをサポートします。また、コマンドラインやより視覚的なグラフィカルユーザーインタフェースの実装も可能です。

2)ほとんどのシステムで使用されている図1の3層モデルは、どんな既定の機能(逆行列プログラムの実行など)をも複数のレベルで提供できることを意味しています。MPI ジョブを実行するバックエンドの並列コンピュータもありますが、ここでは全く異なるコンピュータで動作するいくつかの中間層のコンポーネントによるサービスのフロントエンドとして、異なるセキュリティドメイン上に置くことにします。それぞれのレベルにおけるサービスは対話が可能で、たとえば並列コンピューティングレベルでの高性能の I/O 転送がサービスレベルでの SOAP のようなより低速の中間層プロトコルにより行われます。この 2 つ(以上)のコール(コンポーネントの対話)は異なる機能を提供できますが、中間層コールは高性能なミラーと組み合わせることも可能です。一般に中間層は、制御やバックエンドの「raw データ転送」を提供します。その結果図1のようにやや複雑化したモデルとなっています。図では中間層とHPC(raw)層にはそれぞれのコンポーネント(サービス)が示されています。イントラ層とインター層とのリンクも示されています。参考資料[39]の『Programming the Grid(グリッドのプログラミング)』は、グリッドのさまざまなプログラミングモデルを紹介している優

れた論文です。

3) セキュリティ(認証、許可、プライバシー)は、重要性が広く指摘されている汎用機能であり、基本的にあらゆる環境で、さまざまな方法を用いてこれに対する取り組みが行われています。

4) データ管理は、重要性が幅広く指摘されているもう 1 つのテーマであり、単独のマシンよりも分散システムでの重要性がいっそう高まっています。データ管理には、ファイル操作やデータベース、人工衛星や加速器のような機器から生成される生信号へのアクセスなどが含まれます。

5) 他の汎用ツールのライブラリによる基本的な GCEShell の拡張。これは GCE によるサポートが可能です。こうしたツールには、(グリッド)FTP や(グリッド)MPI、パラメータスイープ、より汎用的なワークフロー、GCEShell プリミティブの構成などがあります。

6) よりハイレベルな他のツールも重要ですが、多くはアプリケーションに依存する傾向があります。ビジュアライゼーションや、使用するアルゴリズムの種類についての知的意思決定支援を、ここに入れることができます。

7) 商用ポータルでは、複数のサブウィンドウを集約した高度なユーザーインタフェースをサポートするのが通例となっています。Apache Jetspeed プロジェクトはこれをサポートしているツールキットとして有名です[33]。このユーザーインタフェースの集約は GCE でサポートすることも少なくありません。この集約については最終の第 5 節で述べます。

GCE は、特定の機能の他に、グリッド専用のコンピューティングモデルを含むのが一般的となっており、このモデルは GCE アーキテクチャ及びユーザーに提示されるグリッドのビューに反映されています。

たとえば、アプリケーションのオブジェクトモデルは広く普及していますが、このオブジェクトビューは、GCE がユーザーに提示するグリッドのビューに反映されています。GCE のプログラミングモデルは、一般に比較的規模の大きいオブジェクトのプログラミングです。プログラムやハードウェア資源は、このオブジェクトモデルがアプリケーションで使用されているソフトウェアモデルを必ずしも変更することなく、オブジェクトとして説明することができます。

以上を前提とすることにより、このドキュメントで引用した論文の分類が可能となります。しかし分散グリッドシステムのような複雑なトピックの場合、絶対的な分類はできないのが常です。したがってこれらのプロジェクトは、多くの重複した視点で検討されることも少なくありません。

3. GCE プロジェクトの概要と機能

3.1 GCE システム構築のための技術

グリッドの構築に必要な基本的なアーキテクチャと技術、そして各種 GCE の基本的なコンポーネントについては、すでに述べたとおりです。また、前掲の参考資料[39]では、バックエンドのプログラミングに関する多くの問題を取り上げています。

Globus ツールキット[32] は、最も広く利用されているグリッドのミドルウェアシステムですが、GCE の構築については十分なダイレクトサポートを提供していません。参考資料 [6, 14, 15, 27] および [44]には、それぞれ Globus ツールキット用の Java, CORBA, Python, Perl による商用グリッドインタフェースが紹介されています。これらは GCE 全体の基本的な構成要素となるものです。参考資料 [1] には Grid Portal Development Toolkit (GPDK) を取り上げています。これは Java ベースの GCE 環境に適した JavaBeans スイートで、Java Server Pages (JSP) 表示のサポートを目的としています。COG キットと GPDK を組み合わせれば、Globus 環境を使用して基本的なグリッドサービスを提供する GCE を構築するという最も広く利用され、フレームワークが確立します。参考資料 [7]、[8]、[20]における問題解決環境は、Java Commodity Grid Kit [6]を基盤とし、また [51]のポータルは、Perl Commodity Grid Kit [27]を直接的基盤としています。

GCE 構築のもう一つの重要な技術に、通知/イベントサービスがあります。参考資料 [21] では、現在、メッセージベースのミドルウェアを基盤とするグリッドアーキテクチャが増え、この傾向が特に Web サービスで顕著になっていることを述べています。このドキュメントは、グリッドのイベントやメッセージングのサポートに関する設計と基本形について説明しています。参考資料 [21,49] は、Narada Brokering システムについて述べています。ピアツーピア技術を使い、広域でメッセージをルーティングするフレームワークを提供するシステムです。GCE がユーザーの環境と対象グリッド間にある信頼境界を通過しなければならない場合、極めて重要な役割を果たします。

参考資料 [9] は、Globus ツールキットとインタフェースをとるための C サポートに関するもので、ツールキットの機能を提供するポータルは本書のインフラストラクチャ上に構築することができます。

す。参考資料 [17] では、学際的なアプリケーションとジオメトリのマッチングの実行時カップリングをサポートする興味深いXML ベースの技術を紹介しています。参考資料 [28] は、比較的異なる技術、つまり新しいスケジューリングアルゴリズム試験を目的としたグリッドシミュレータについて述べています。

3.2 広範な問題解決環境

GCE が提供するユーザーインターフェースは、概ね次の2つのクラスに分類されます。1つは特定のアプリケーション(セット)に焦点を当てたクラスで、アプリケーションポータルまたは問題解決環境 (Problem Solving Environment: PSE)とも呼ばれています。もう1つは、包括的なアプリケーション機能を提供するもので、ユーザーポータルと呼ばれています。これまで紹介してきた表記に基づき、これを GCEShell ポータルと呼ぶことにします。実際、GCEShell ポータルをベースとした PSE で階層が構築されるケースが多いようです。GCEShell ポータルは GPKD のようなミドルウェアを、また GPKD は Java CoG Kit [6]を基盤とし、Java CoG Kit 自身は Globus ツールキットを、Globus ツールキットはグリッドコンポーネント資源のネイティブ機能を基盤としています。この階層は 1 組の技術とアーキテクチャを対象としたものですが、他のアプローチも同様に階層的な方法で構築されています。

他の論文にもグリッドの PSE に関する議論が掲載されています。参考資料 [5] は、グリッドポータルアプローチへの転換に伴う、「レガシー」PSE に対するアーキテクチャの変更に関する興味深い議論を掲載しています。参考資料 [11] では、複数のオペレーションシステムの調査に基づいて、PSE の機能の豊富さについて紹介しています。これらは参考資料 [16] の PSE と共通財産を共有していますが、後者の論文は主に後で述べる推奨ツールに焦点を当てています。

さらに 5 つの論文では、使用されている GCE インフラストラクチャや研究されているアプリケーションの観点から見て異なる PSE について述べています。参考資料 [7]は、オブジェクトコンピューティングモデルを使った GCEShell ポータルをベースとする 2 つの PSE について説明しています。同様のポータルには XCAT Science ポータル [29]があり、Web ページや Python スクリプト、アプリケーション専用の制御コードを持つアプリケーションノートブックの概念に基づいています。このケースでは、Python スクリプトコードは GCEShell シェルの役割を果たします。天体物理学の共同実験室[20]では、Java [6] と GPKD [1]を介した Globus ツールキットリンクが使われています。また、強力な Cactus 分散環境[31]とインターフェースが取られています。参考資料 [18] と [47] は、Web サービスを使用した計算物理、特にデータ操作サービスのポータルを紹介しています。Polder システム [24] と SCIRun [25]は、生物医学を含むいくつかのアプリケーション内で、豊富な視覚化機能を提供します。SCIRun は NetSolve [10]を含むいくつかのグリッド技術にリンクされており、強力

なワークフロー機能を備えたコンポーネントモデル(参考資料 [53]に記載の CCA [34])をサポートします。

参考資料 [48] の検出システムは、リモートグリッドアプリケーションのコンピュータ操作を行うために構築された PSE フレームワークについて述べています。これは、参考資料 [42]の『Classifying and enabling Grid applications(グリッドアプリケーションの分類と利用)』で説明している Cactus での重要な作業目標でもあります。

3.3 広範な基本 GCEShell ポータル

ここでは、グリッドの汎用的なコンピューティング機能をサポートするために設計された一組のポータルについて説明します。参考資料 [3] は、DoE の ASCI プログラムの厳密な要件をサポートするために設計されたグリッドポータルを取り上げていて、興味深いものです。これはセキュリティやパフォーマンスの問題だけでなく、ASCI マシンを使う計算物理学者向けの実績あるコンピューティングモデルも反映しています。参考資料 [4] は、極めて高度な Legion Grid に対応するポータルインタフェースについて説明したものです。Legion Grid は、Legion Shell を通じて Legion [43]でサポートする共用オブジェクト(ファイル)システムに強力な汎用インタフェースを提供します。本書では、基本 GCEShell ポータルのトピックをベースに、どのように特定の問題解決環境を構築したらよいかを解説しています。

Unicore [23] はもともとヨーロッパの専用スーパーコンピュータへのアクセスをサポートするために開発されたフル機能の GCEShell ポータルの 1 つでしたが、最近では Globus ツールキットや、参考資料 [50]で述べているように、オープングリッドサービスアーキテクチャ[56]とインタフェースをとるようになっていきます。

Unicore は、完全なワークフローのサポートにより、興味深い抽象的なジョブオブジェクト (Abstract Job Object: AJO)を開発しています。

参考資料 [7, 13, 45] は、いくつかのアプリケーション専用の PSE が構築され、入念に開発された GCEShell ポータル技術について述べています。参考資料 [51] は、NPACI Grid Portal ツールキット、つまり Perl Community Grid Kit [27] を使って Globus ツールキットにアクセスするためのミドルウェア、GridPort について解説しています。参考資料 [26] も HotPage、GridPort をベースに構築された GCEShell について説明しています。

3.4 ワークフロー

ワークフローは、複数の分散コンポーネントから1つの完全なジョブを構成することに相当します。これは重要性が広く認識されており、商用 Web サービスのコミュニティでは主要なテーマとなっています。また、これらのシステムはタスクの特定シーケンスの構成物であるため、本来は GCEShell または PSE の一部と見ることもできます。このテーマには複数のプロジェクトが取り組んでいますが、今のところワークフローの表し方について一致した意見は見られません。ただ、いくつかのグループがコンポーネント間の結合を定義するため、視覚的なインタフェースを開発しています。ワークフローについては、論文の[3], [8], [17], [23], [25]で議論されています。後者はグリッドのワークフローをデータフローパラダイムと統合しています。これはビジュアライゼーションコミュニティで実績があります。BPEL4WS は、グリッドのコミュニティに大きな影響を与える可能性のある新しいワークフローの重要な標準化提案[60]です。参考資料 [17] は、アプリケーションのカップリングをサポートするための強力なランタイムの必要性を強調していますが、参考資料 [8]他の論文では示唆にとどまっています。本件については第 4.2 節で詳しく説明します。

3.5 データ管理

データ集中型のアプリケーションは、グリッドでは極めて重要と考えられていますが、そのサポートに関しては、ここでは取り上げていません。GridFTP のようなメカニズムを介してのファイルシステムやデータベース、データ転送とのインタフェースについては、いくつかの論文でカバーされています。これは基本的に、グリッドではデータ管理ソフトウェアが未だに比較的新しい存在であるためです。参考資料 [47] では、SOAP ベースの Web サービスや、大規模な科学データグリッドプロジェクト内で使用するデータを管理する、ポータルインタフェースについて解説しています。

今日のグリッドポータルのは大半は、ユーザーがポータルを利用してファイルのアップロードやダウンロード、任意の場所への移動などが行えるように、GridFTP コンポーネントを装備しています。

3.6 GCEShell ツール

GCE コンピューティングモデルでは、基本的な GCEShell 機能に新たな価値を付加するツールのライブラリを構築することを考えます。先の 2 つのサブセクションでは、特に関心の高いワークフローとデータ管理の 2 つのツールを紹介しました。ここでは、「Grid Computing Environments special issue(グリッドコンピューティング環境特別号)」に掲載されたいくつかの論文で紹介されている他のツールを幅広く紹介します。

Netbuild [2] は、グリッドの異種性が高まる中、広範なターゲットマシンを対象にソフトウェアの自動構成機能を発揮する分散ライブラリをサポートします。NetSolve [10, 46] は、適切なグリッド資

源をクライアントニーズにマッピングするためのエージェントを初めて採用しました。参考資料 [16] では、推奨システムについて述べています。ここでは詳細なパフォーマンス情報をもとに、PSE のユーザーを支援し、問題の解決に役立つ最良のアルゴリズムを選択します。

多くのプロジェクトでは、既定のアプリケーションが異なる入力パラメータで何度も実行されるといふ、「パラメータスイープ」の問題の重要性を指摘しています。このような問題はグリッド環境にとって最適であり、参考資料 [22]では特定のパラメータスイープシステム、Nimrod-G について解説しています。この論文では、異なるツール、つまりグリッドサブリヤやコンシューマによる経済的モデルをベースとした斬新なスケジューリングツールに焦点を当てています。参考資料 [40] は、もう 1 つのパラメータスイープシステムとして有名な APST について取り上げています。APST は、AppLeS アプリケーションレベルのスケジューリングシステムをもとにしています。

参考資料 [26 と 51]で説明されている HotPage は、ポータルへのジョブの状態情報を供給する初めてのツールとして有名です。このようなツールは明らかに広く重要性が認められています。

多くのユーザーにとって重要なツールに、ビジュアライゼーションがあります。これについては参考資料 [25] と [17]で取り上げられています。他にも高度なグリッド分析機能のカテゴリに該当するデータマイニングのような重要なツールも多数あります。

この領域については、前に紹介したように基本的なグリッド「プログラミングプリミティブ」が「GCE Shell」と表される UNIX からよく使われているアイデアを拝借することで、秩序だてることができると思われます。上に述べたように、Shell プリミティブは、異なるパラダイムやその表現を用いて、さまざまな方法でユーザーに供給されます。Shell プリミティブの供給方法の 1 つはコマンドラインインタフェースですが、多くの場合、より上位のビューが提示されます。完全なドメイン専用の上位システムは、上に述べた問題解決環境です。Legion Grid システム [4]は、UNIX から自然に拡張する Legion シェルを用いて GCE Shell を明快に説明しています。GCE Shell には UNIX Shell と共通する機能がいくつかあります。たとえば、ファイル操作は UNIX とグリッドの両方にとって極めて重要です。また一方では興味深い違いも見られます。たとえばグリッド(および GCE Shell)では次の表現が求められます。

サービスとユーザー間のネゴシエートされた対話

システムの全レベル(ローカルクライアント、中間層、バックエンド資源)におけるファイルとサー

ビス

オブジェクトとそのメタデータとの差異。オブジェクトのコピーは主要なハイパフォーマンスタスクだが、メタデータのコピーは一般に中程度の作業である

必要とされるプリミティブを調べると、UNIX Shell に比べ、GCE Shell では次のような機能を追加する必要があります。

検索

検出

登録

セキュリティ

UNIX Shell の pipe や tee より高度なワークフロー

JXTA [35]にあるようなグループその他のコラボレーション機能

メタデータの処理

管理とスケジューリング

ネットワーク

サービスの対話のためのネゴシエーションプリミティブ

GCE Shell について考える場合、ファイルや実行可能ファイルがいずれもサービスであり、UNIX のように区別がされていない画一的なモデルを使って議論を簡単にすることができます。ここでは、個々のファイルアクセスが、異なった実装が可能であったとしても Shell ではサービスであるという「バーチャルサービス」の概念が必要です。これは新しいコンパイラ研究で考えられる領域の一つです。

GCE Shell の本質は、グリッドのプログラムに必要なプリミティブ機能のカatalogに過ぎません。

実際、上のリストは OGSA の一部であるコアサービスのサブセットです。グリッドのプログラミングパラダイムは、アプリケーション構築のためにコアサービスを操作する固有の方法です。第 3.3 節および 5 節で述べているポータルサービスは、ユーザーと対話するための方法です。これらを総括したものが、第 3.2 節で述べた問題解決環境となります。

4. GCE コンピューティングモデル

4.1 GCE モデルに関する調査

前文で、グリッドコンピューティング環境の基礎となるコンピューティングモデルを考えることは興味深いと述べました。これは、ファイルやコンピュータ、データベース、ポータルを通じて提供されるプログラムの世界について考察する方法に言及したものです。

参考資料 [10 と 46]の NetSolve は、参考資料 [30 と 41]の Ninf 研究とともに、分散コンピューティングのための重要なネットワークサービスモデルを開発しました。

各ユーザーが問題の一部を解決するためライブラリをダウンロードする代わりに、このタスクがネットワーク資源に配送され、コンピュータを使用したサービスを提供します。

Ninf および NetSolve はどちらも、新しい GridRPC リモートプロシージャコール標準をサポートし、参考資料[39]で説明されているグリッドコンピューティングモデルの主要な中核機能をカプセル化します。GridRPC は、科学的データ構造やグリッド固有のセキュリティおよび資源管理をサポートします。

Raw (HPC) Resource	raw(HPC) 資源
Application Software	アプリケーションソフトウェア
HPC Facing Ports	HPC 向けポート
“Middle-Tier”	「中間層」
Application Service	アプリケーションサービス
Service Facing Ports	サービス向けポート
User Facing Ports	ユーザー向けポート

図 2. ユーザーとのやり取り(ポータルインタフェース)、プロキシーとraw資源、その他の中間層コンポーネント間、その他の raw(HPC)資源間の、4つのタイプの対話を表すプロキシーサービスプログラミングモデル。

参考資料 [12] はMPI(並列コンピューティングのメッセージパッシング基準)のグリッド実装を表したもので、異なる並列コンピュータに関してMPI実装とバイナリ表現間の非互換性の問題を解決します。なお図1の表記では、MPIはミドルウェア層ではなく「HPC バックエンドリンケージ」層にある

ことに注意してください。参考資料 [20]では Cactus 環境 [31, 42]をサポートしています。Cactus 環境は、HPC 層のグリッドコンピューティング向けに入念に開発されたもので、中間層の GCEShell 環境ではなくバックエンドのプログラミングインタフェースをサポートします。参考資料 [20]の天体物理学の問題解決環境では、完全な中間層環境により Cactus を拡張しています。

参考資料 [4, 43]で述べている Legion は、極めて完成度の高いグリッドオブジェクトモデルを構築しました。参考資料 [8] はグリッド用の CORBA 分散オブジェクトモデルを取り上げ、参考資料 [19 および 48] では、複数の CORBA GCE 間に相互運用性を提供するにあたっての非常に困難な問題に言及しています。CORBA は、参考資料 [14]に記載したようないくつかの非互換実装とともに発展してきたものですが、使用する技術(XML, SOAP)は CORBA よりもオープンであることから、Web サービスは容易に相互運用性を実現できると期待できるでしょう。

参考資料 [7, 13, 23, 45, 50] と XCAT サイエンスポータル [29, 53] もまた、GCE コンピューティングのオブジェクトモデルを表していますが、1 つの重要な機能、つまり中間層オブジェクトは常にプロキシであり、これが、従来環境で動作する「真の資源」を表すメタデータを保持しています。このプロキシ戦略は多くのグリッド資源にとって有益であるかのように見えますが、NetSolve の真のネットワークサービスモデルも同様に不可欠です。ここで UNIX の例として、(異なるマシンにある)2 つのプログラムの間でデータをやり取りする場合を考えてみましょう。

ここではプログラム内のメカニズムを選択しますが、単純なソケット、FTP、RMI インタラクションメカニズムのいずれかを使用します。

あるいは、プログラムを出力と入力、または「標準 I/O」で汎用的に記述することも可能です。次にこのプログラムは、UNIX Shell コマンドによるもう 1 つの入力に対して 1 つの「pipe」された出力を持つことができます。内部的に指定されたアクションと、サービスレベルで指定されたアクションを持つようなハイブリッドプログラミングモデルは、グリッドの成功にとって重要であり、グリッドのためのプログラミングモデルに組み込まれるべきものと考えます。

どのような GCE コンピューティングモデルも、メタデータのみと、グリッドオブジェクトのラップスタイルの両方をサポートしなければなりません。事実、第 2 節のポイント 2)に戻ると、プロキシと NetSolve モデルは図 2 や図 3 で示したほど実際の差異はありません。2 つのモデルは効果的にアプリケーション(ソフトウェア)資源をオブジェクトとしてラップします。プロキシモデルでは、中間層とバックエンド間の対話を提供します。NetSolve および Ninf のラップサービスモデルでは、ユーザーに 1 つのエンティティを提供します。いずれの場合も、個々の中間層および HPC(「真

の) コミュニケーションを持つことができます。分類をより複雑にすれば、当然ながらプログラミングモデルの抽象概念(プロキシーか否かによらず)と実装との間には差異が生じることになります。XCAT モデルではソフトウェアコンポーネントシステム [53] が使用され、ラップサービスまたはプロキシーモデルを実装します。コンポーネントシステムは Web サービス標準をベースとしているため、ラップサービスコンポーネントを任意のグリッドサービスとすることも可能です。

コンピューティングモデルにおいて、他に GCE システムで解決しなければならないのは、資源の管理方法です。参考資料 [22, 52] の『Grid Resource Allocation and Control Using Computational Economies (計算経済を利用したグリッド資源の割当てと管理)』で、著者は資源の割当てとプロビジョニングの経済学的モデルの事例を作成しています。グリッドシステムが極めて大規模になるにつれて、このようなアプローチが使われるようになる可能性は高いと考えられます。

Raw (HPC) Resource	raw (HPC) 資源
Application wrapped	アプリケーションラップ
HPC Facing Ports	HPC 向けポート
“Middle-Tier”	「中間層」
as a Service	サービスとして
Service Facing Ports	サービス向けポート
User Facing Ports	ユーザー向けポート

図 3. ユーザーとのやり取り(ポータルインタフェース)、他の中間層コンポーネントとのやり取り、その他の raw(HPC) 資源間の、3 つのタイプの対話を表すラップアプリケーションプログラミングモデル

4.2 2 層プログラミングモデル

アプリケーションソフトウェアを単純な 2 つの階層で考えることにより、GCE コンピューティングモデルの議論が進めやすくなります。個々の CPU を制御する「マイクロスコープ」ソフトウェアがあり、Fortran や C++、Python などの一般的な言語で記述されています。そしてこれらの言語は「ナゲット」、つまりコードモジュールを生成します。「従来型」のプログラミングでは、これらのナゲットをある 1 つの資源に関連付けています。ナゲットは、たとえばデータベースに対する SQL インタフェースや並行画像処理アルゴリズム、有限要素ソルバなどが考えられます。(解決には至っていないものの)深く理解されているこの「ナゲットプログラミング」は、分散ナゲットをすべて完全な「実行

ファイル」に統合することで、グリッド用に拡張しなければなりません。

ナゲット内部のプログラミングは、現在グリッドの範疇外と考えられています。ただし、GrADS [57] のようなプロジェクトでは個々の資源(ナゲット)やグリッドプログラミングの統合について研究を行っています。ここでは、それぞれのナゲットがすでにプログラムされており、その統合だけを考えればよいと仮定します。この統合とは、実際にはごくありふれたもので、UNIX オペレーティングシステムの単一の資源で使われている「Shell/Perl...」スクリプトや、PC Case における Microsoft Com/ActiveX/... インタフェースなどの使用が一般的です。

このスタイルのグリッドプログラミングはいくつか現れはじめています。主要なものは第 3.2 節の問題解決環境で、入念に選択されたツールセットや、通常特定の問題ドメイン向けにカスタマイズされたアプリケーションサービスに対するポータルインタフェースを特徴としています。これには、第 5 節で述べているグラフィカルユーザーインタフェースや、PSE の異なる部分をリンクする「ソフトウェアバス」といったものも含まれます。

アプリケーションナゲットの統合(またはソフトウェアバス)は、一般に「ワークフロー」と呼ばれ、第 3.5 節で述べているように、これを表現するための多くの異なるパラダイムがユーザーに提案されています。

1 つの一般的なモデルは、ナゲットをパレットから選択し、その「ポート」またはチャンネルをリンクできるグラフィカルインタフェースです。これはビジュアライゼーションや画像処理で普及しているもので、AVS [58] や Khoros [59] などのシステムが確立されています。業界は、BPEL4WS (Business Process Execution Language for Web Services [60]) や WSCL (Web Services Conversation Language [61]、会話するのはユーザーではなくナゲット)のようなアプローチを用いて、このナゲットのリンケージのための XML 仕様を開発しました。より簡単で、おそらくより強力なのは、(Python のような)スクリプティングや(Java などの)コンパイル型言語を使ってリンケージをプログラミングする「だけ」です(これがソフトウェア「バス」の考え方)。複数のパラダイムや複数の言語をサポートすることは有用であると期待できますが、これらのいずれかが「最良」とは考えられません。ナゲットのプログラミングを表す重要なグリッドのアプローチには、DoE の CCA (Common Component Architecture [34, 53]) や UK e-Science Program の ICENI プロジェクト [62] などがあります。

上の例の趣旨をまとめると次のようになります。「グリッドのユーザービューのプログラミング」は

(分散)オブジェクト技術と重複していますが、このドキュメントでは「特定のプログラミングモデルを推奨」しようとしているのではなく、「取り組むべき問題」を説明しようとしたものです。そして実装における差異の大きさよりも、取り組むべき問題の共通性について強調しています。PC やワークステーションのプログラミングでおなじみのタスクと同類ですが、「グリッドのユーザービュアのプログラミング」の方は極めて複雑です。図 1 で解説したように、「実行可能ファイル」(統合ナゲット)は、システムとアプリケーションサービスを混合したものです。システムサービスは単一のワークステーションで使用しますが、現在のところグリッドのメタ OS サービスはプログラム可能なインタフェースを装備するものと期待されています。他方、多くの対応ワークステーション(Windows, UNIX) サービスは依然不透明です。図 1 の多くのシステムサービスが OGSA 構想の一環として定義および実装されるようになれば、OGSA は図の構成要素となります。

システムやアプリケーションナゲットが豊富なだけでなく、多くのグリッドシステムは個々に、「実際」のエンティティ(ソフトウェアナゲットなど)と「実際」のエンティティを表現するメタデータを表すエンティティの両方を維持します。より多くのメタデータの定義が必要なのは明らかなので、この個別化は今後も続き、実際に使用が拡大するものと考えられます。また、このメタデータがそれ自身の表現する資源とは別に格納されることも少なくないと思われる。

典型的なナゲットプログラミングの課題として、ナゲット間の最も適切な通信機構を決定するために、アプリケーションで必要とされるレイテンシ/帯域幅と、ネットワークの制約(ファイアウォール)の 2 つの点を考慮しなければなりません。特定のサービス間対話の実装における代表的なランタイム仕様には、合意されたアプローチはありません。特定の实装方法による使用例が多々あることは確かです。「エージェント」や「ブローカー」、「プロファイル」は、この適応機構を記述するためによく使用される言語の典型です。事実、エージェントのフィールドは、グリッドのフィールドとの統合も可能と考えられます。グリッドプログラミングの開発においては、次の 2 つを研究する必要があります。

プログラミングパラダイムおよびパラダイム内では特定の言語を選択 - スクリプト化、ビジュアル、コンパイル可能

ランタイムライブラリ、主として機能性において異なるパラダイム間で共有可能だが、各個別のアプローチでは比較的異なる表現で表されるもの

このドキュメントで紹介した記事の多くは、問題のどちらの側面を強調するかで、部分的な差異が見られます。

Resource-facing Ports	資源向けポート
WSDL	WSDL
Content Provider	コンテンツプロバイダ
Web Service	Web サービス
User Facing Ports	ユーザー向けポート
Other WS	その他の WS
Other	その他
Portal	ポータル
Aggregate WS-User Facing Fragments	
WS-ユーザー向けフラグメントのアグリゲート	
Render	レンダー

図 4: コンテンツ提供 Web サービスのユーザー向けポートで生成されたドキュメントのフラグメントのアグリゲーションサービスを提供するポータル

5. ポータルサービス

ポータルサービスは、ユーザーインタフェースと対話の管理と提供を担うサービスです。図 4 はこの領域における主なアーキテクチャの概念を示したものです。ユーザーに提示されるすべての素材は、ここではコンテンツプロバイダと呼ぶ Web サービスから発信されるものとします。このコンテンツは、シミュレーションやデータリポジトリ、機器のストリームによって生成されます。このような Web サービスは、それぞれ資源またはサービス向けポート(図 4 では RFIO)を備えており、いずれも他のサービスとのコミュニケーションに使用されます。ここでは、ユーザー向けのコンテンツを生成し、クライアントデバイスからの入力を受け入れるユーザー向けポートに注目してみましょう。これらのユーザー向けポートは、OASIS 団体によって標準化されている WSDL の拡張を使用しています。これは WSRP (Web Services for Remote Portals : <http://www.oasis-open.org/committees/wsrp/>)と呼ばれています。WSRP はいわゆるポートレットインタフェースを実装していますが、これは JCP (Java Community Process) の一環として Java で標準化されています。

ほとんどのユーザーインタフェースでは、1 つ以上のコンテンツプロバイダからの情報を必要とします。たとえば、コンピューティングポータルには、ジョブの提出、ジョブの状態、ビジュアライゼーション、その他のサービス用の個々のパネルを特徴とするものがあります。これをカスタムアプリケ

ーション固有の Web サービスに統合することもできますが、汎用のアグリゲーションサービスを提供の方が魅力的です。これによりユーザーや管理者は、表示するコンテンツプロバイダや、表示の占有部分を選択することができます。このモデルでは、各コンテンツプロバイダは、ポータルによって統合される各自の「ユーザー向けドキュメントのフラグメント」を定義します。このようなポータルの集約は、主要なコンピュータベンダーや、有名な Jetspeed プロジェクト (<http://jakarta.apache.org/jetspeed/>) では Apache によって提供されます。ポートレットは、Web サービスがミドルウェアのコンポーネントモデルを表すのと同じ方法で、ユーザーインターフェースのコンポーネントモデルを表します。このアプローチを利用することにより、再利用性やモジュール性の利点が得られることは明らかです。中間層のコンポーネント(ナゲットを表現する Web サービス)を統合するワークフローや、ユーザーインターフェース用にこれらを統合するポータルの集約では簡潔な表示が得られます。図 5 は、Gannon と Plale 主導のプロジェクトで、NCSA Alliance 用に開発されたポータルを用いてこの考え方を表したものです。ここには、異なる GCE Web サービスに対してそれぞれ 4 つのインターフェース(左側に 3 つ、右側に 1 つ)が見られます。個々の Web サービスは、1 つ以上の GCEShell 機能と関連付けることができます。それ以上の機能は最上位でタブを使って集約されます。このプロジェクトでは、統合を容易にするコンポーネントインターフェースアーキテクチャを用いて特定のユーザーインターフェースのフラグメントを開発している、様々な団体が多数関与しています。異なるグループの作業の集約化は、中間層の Web サービス(OGSA)と、ユーザーインターフェースでポートレットを体系的に使用することで行われます。

図 5: 複数のコンピューティングサービスに対してインターフェースの集約機能を持つ日本語ポータルの例

図 6 は、表示したコンテンツを特定のクライアントに適応させる機能に対応するいくつかの他のポータルサービスについて示したものです。ここではデバイス間(没入型、デスクトップ型、ハンドヘルド型など)の差異とユニバーサルアクセスの問題の両方に対処し、ユーザーの考えられる物理的限界に適応しています。図 4 のアーキテクチャは、レンダリングされるビューを定義するためのクライアントとコンテンツプロバイダ間の交渉が必要であるため、より複雑になります。これには、ユーザープロファイルを処理し、適切なコンテンツを選択するポータル選択サービスが必要です。また、たとえばマルチメディアコンテンツの解像度を圧縮するための共通のフィルタをパッケージングすることも可能です。ユニバーサルアクセスに関するこの研究は、オーディオ-ビデオ会議システム(クライアントに適合させるために H323 のようなプロトコルで「ベスト」コーデックをネゴシエート)では一般に行われており、アクセシビリティ構想の一環として W3C によって究明されています。アグリゲータ、セクター、フィルタリングといった一連の機能は、複数のグリッドアプリケーションで共有可能な共通のポータルサービスを表しています。

5.1 GCE ポータル対応のオープングリッドサービスアーキテクチャの実装

OGSA は、オープングリッドサービスインフラストラクチャ(OGSI)仕様の見地から定義された一連のコアグリッド Web サービスで構成されています。OGSI 準拠のグリッド Web サービスは、ポートをすべて標準のグリッドサービスポートから継承したグリッド Web サービスのサブクラスを定義します。このポートを使用した標準的な方法では、リモートポータルがサービスに応答指令信号を送信することにより、サービスが実装する他のポートタイプや、こうしたポートで可能なオペレーション、サービスの共用内部状態などを検知することができます。OGSI サービスはまた、単純なイベントの予約や通知方法を標準的な方法で実装することもできます。さらに、サービスを、サービスのコレクションにグループ化する仕組みも提供します。OGSI の簡素で標準的な特性により、実行時コンパイラの構築が可能となり、OGSI 準拠のあらゆるグリッドサービスに対応するポータルポートレットインタフェースを作成することができます。

OGSA で定義されているコアサービスには、レジストリ、ディレクトリ、名前空間束縛、セキュリティ、資源の記述、資源サービス、予約とスケジューリング、メッセージングとキューイング、ロギング、アカウントिंग、データサービス(キャッシュおよびレプリカマネージャ)、トランザクションサービス、ポリシー管理サービス、ワークフロー管理およびアドミニストレーションサービスなどがあります。こうしたコアサービスは、それぞれグリッド Web サービスとして提供されます。(本ドキュメント執筆時点のこのリストは不完全であり、正式なものではありません。)OGSA 準拠のグリッド向けのアプリケーションでは、これらのサービスが利用可能で、適正な承認を受けることにより、使用できると見てよいでしょう。

我々は、標準 GGF-GCE ポータルフレームワークの一環として、こうしたサービスに対応するクライアントおよび管理ポートの両方にアクセスするためのポートレットを構築および配布することができます。OGSA は、GCE ポータルおよびアプリケーションのための、自然で使いやすい構成プラットフォームを提供します。

6. セキュリティ要件

グリッドポータルの基本的な役割の1つは、グリッド資源への安全なアクセスを管理することです。したがって、ほとんどの論文でもこのテーマに関する議論が見られます。Globus と Legion をベ-

スとする GCE は、PKI(公開鍵基盤)を使用しています。一部のインストレーション(米国の場合は DoD や DoE など)には Kerberos が必要であり、これらのために開発されたグリッドコンピューティング環境[3] [7] [13]は、このセキュリティモデルをもとにしています。

今日の多くのポータルでは、ユーザーのブラウザとポータルサーバー間に HTTPS コネクションを使用しています。

しかし、身分証明書は通常、「MyProxy」サーバーとして知られる信用ある第三者に格納されます。ユーザーは一日～数日有効なプロキシー証明を、個人のパスワードを使って MyProxy サーバー上に格納しなければなりません。この後ポータルサーバーはユーザーにパスワードを要求し、次に MyProxy サーバーに対してプロキシー証明をコピーするように求めます。ポータルにプロキシー証明がコピーされると、今度はポータルに対して、リモートグリッドサービスとの対話時にユーザーの代わりにこれを使用するよう託します。

セキュリティは、GCE ポータルにとって基本的な重要性を持つため、我々は引き続き GGF OGSi セキュリティワーキンググループの進展を見守っていく所存です。

7. 著者の連絡先

Geoffrey Fox
Community Grid Computing Laboratory,
Indiana University
501 N Morton Suite 224
Bloomington IN 47404
gcf@indiana.edu

Marlon Pierce
Community Grid Computing Laboratory,
Indiana University
501 N Morton Suite 224
Bloomington IN 47404
marpierc@indiana.edu

Dennis Gannon

GFD-I9

Department of Computer Science
Lindley Hall 215
Indiana University
gannon@cs.indiana.edu

Mary Thomas
Texas Advanced Computing Center, The
University of Texas at Austin, 10100 Burnet
Road,
Austin, Texas 78758
mthomas@tacc.utexas.edu

8. 知的所有権表示

GGF は、いかなる知的所有権、本書に述べられている技術の使用、実装に関わるものとして主張され得るその他の権利の有効性や適用範囲、あるいはこのような権利のもとでライセンスが利用可能か利用不可能かの範囲に関しては、いかなる立場もとりません。また、そうした権利を特定するために取り組んできた事実はありません。出版のために利用できる権利の主張のコピーや利用されるライセンスのいかなる保証、この仕様の実装者またはユーザーによるそうした所有権の使用に対して、一般的なライセンスまたは許可を得るための手続きの結果は、GGF 事務局より入手することができます。

GGF は、すべての利害関係者に対して、この推薦内容を実践するために必要な技術を保護している可能性のある著作権や特許、特許の申請、その他の所有権に対し、注意を払われるようお勧めします。詳細は GGF エグゼクティブディレクタ宛にお送りください。

9. 著作権全文表示

Copyright (C) Global Grid Forum (2002). All Rights Reserved.

上記著作権表示と本パラグラフが全ての複製文書や派生的な研究に含まれている限りにおいて、

いかなる種類の制限を課すことなく、一部または全部を対象に、この文書およびこの文書の翻訳物を複製し他者に供することができます。また、その内容に対するコメントや説明、あるいはその実施を支援する派生物を作成、複製、発行、配布することができます。しかしながら、この文書自体は、グリッドの推薦内容を発展させる上で必要とされる場合(この場合は GGF 文書プロセスに定義された著作権の手続きに従わなければならない)、あるいはそれを英語以外の言語に翻訳する必要がある場合を除き、著作権表示や GGF その他組織への照会情報を削除するなど、いかなる方法によっても変更することはできません。

上記に規定する限定的な許可は永続的であり、GGF またはその後継者や譲受人によって無効となることはありません。

この文書およびこの中に記載された情報は「現状有姿」で提供されます。グローバルグリッドフォーラムは、ここに含まれる情報の利用がいかなる権利をも侵害しないことの保証、商業性あるいは特定目的への適合性のいかなる黙示的な保証など、明示的、黙示的を問わずすべての保証を放棄します。

10. 参考資料

- 1) Jason Novotny, 『The Grid Portal Development Kit』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1129-11444 ページ、(2002)。この論文は参考資料 [37]の 657-674 ページ、第 27 章にも掲載。
- 2) Keith Moore and Jack Dongarra, 『NetBuild: Transparent Cross-Platform Access to Computational Software Libraries』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1445-1456 ページ、(2002)
- 3) Randal Rheinheimer, Steven L. Humphries, Hugh P. Bivens and Judy I. Beiriger, 『The ASCI Computational Grid: Initial Deployment』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1351-1364 ページ、(2002)
- 4) Anand Natrajan, Anh Nguyen-Tuong, Marty A. Humphrey and Andrew S. Grimshaw, 『The Legion Grid Portal』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1365-1394 ページ、(2002)
- 5) Karen Schuchardt, Brett Didier and Gary Black, 『Ecce - A Problem Solving Environment's Evolution Toward Grid Services and a Web Architecture』、Concurrency and Computation:

- Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1221-1240 ページ、(2002)
- 6) Gregor von Laszewski, Jarek Gawor, Peter Lane, Nell Rehn, and Mike Russell 『Features of the Java Commodity Grid Kit』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1045-1056 ページ、(2002)
 - 7) Tomasz Haupt, Purushotham Bangalore and Gregory Henley, 『Mississippi Computational Web Portal』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1275-1288 ページ、(2002)
 - 8) Andreas Schreiber, 『The Integrated Simulation Environment TENT』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1553-1568 ページ、(2002)
 - 9) Giovanni Aloisio and Massimo Cafaro, 『Web-based access to the Grid using the Grid Resource Broker Portal』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1145-1160 ページ、(2002)
 - 10) D. Arnold, H. Casanova, and J. Dongarra, 『Innovations of the NetSolve Grid Computing System』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1457-1480 ページ、(2002)
 - 11) Naren Ramakrishnan, Layne T. Watson, Dennis G. Kafura, Calvin J. Ribbens, and Clifford A. Shaffer, 『Programming Environments for Multidisciplinary Grid Communities』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1241-1274 ページ、(2002)
 - 12) M. Mueller, E. Gabriel and M. Resch, 『A Software Development Environment for Grid Computing』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1543-1552 ページ、(2002)
 - 13) Marlon. E. Pierce, Choonhan Youn, and Geoffrey C. Fox, 『The Gateway Computational Web Portal』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1411-1426 ページ、(2002)
 - 14) Gregor von Laszewski, Manish Parashar, Snigdha Verma, Jarek Gawor, Kate Keahey, and Nell Rehn, 『A CORBA Commodity Grid Kit』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1057-1074 ページ、(2002)
 - 15) Keith Jackson 『pyGlobus: A Python interface to the Globus Toolkit』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue

- 13-15, 1075-1084 ページ、(2002)
- 16) E. Houstis, A. C. Catlin, N. Dhanjani and J. R. Rice, N. Ramakrishnan and V. Verykios, 『MyPYTHIA: A Recommendation Portal for Scientific Software and Services』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1481-1506 ページ、(2002)
 - 17) erry A. Clarke and Raju R. Namburu, 『A Distributed Computing Environment for Interdisciplinary Applications』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1161-1174 ページ、(2002)
 - 18) William A. Watson III , Ian Bird, Jie Chen, Bryan Hess, Andy Kowalski and Ying Chen, 『A Web Services Data Analysis Grid』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1303-1312 ページ、(2002)
 - 19) Vijay Mann and Manish Parashar, 『Engineering an Interoperable Computational Collaboratory on the Grid』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1569-1594 ページ、(2002)
 - 20) Gregor von Laszewski, Michael Russell, Ian Foster, John Shalf, Gabrielle Allen, Greg Daues, Jason Novotny and Edward Seidel, 『Community Software Development with the Astrophysics Simulation Collaboratory』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1289-1302 ページ、(2002)
 - 21) Geoffrey Fox and Shrideep Pallickara, 『An Event Service to Support Grid Computational Environments』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1097-1128 ページ、(2002)
 - 22) Rajkumar Buyya, David Abramson, Jonathan Giddy, and Heinz Stockinger, 『Economics Paradigm for Resource Management and Scheduling in Grid Computing』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1507-1542 ページ、(2002)
 - 23) Dietmar W. Erwin, 『UNICORE A Grid Computing Environments』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1395-1410 ページ、(2002)
 - 24) K. A. Iskra,R. G. Belleman, G. D. van Albada, J. Santoso, P. M. A. Sloot, H. E. Bal, H. J. W. Spoelder and M. Bubak, 『The Polder Computing Environment: a system for interactive distributed simulation』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1313-1336 ページ、(2002)

- 25) Chris Johnson, Steve Parker and David Weinstein, 『Component-Based Problem Solving Environments for Large-Scale Scientific Computing』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1337-1350 ページ、(2002)
- 26) M. Thomas, M. Dahan, K. Mueller, S. Mock, C. Mills, and R. Regno, 『Application Portals: Practice and Experience』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1427-1444 ページ、(2002)
- 27) S. Mock, M. Dahan, M. Thomas and G. von Lazewski, 『The Perl Commodity Grid Toolkit』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1085-1096 ページ、(2002)
- 28) Manzur Murshed, Rajkumar Buyya, and David Abramson, 『GridSim: A Toolkit for the Modeling and Simulation of distributed resource management and scheduling for Grid Computing』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1175-1220 ページ、(2002)
- 29) S. Krishnan, R. Bramley, M. Govindaraju, R. Indurkar, A. Slominski, D. Gannon, J. Alameda and D. Alkaire 『The XCAT Science Portal』、Proceedings SC2001, 2001 年 11 月、Denver.
- 30) Ninf network server project <http://ninf.apgrid.org/>
- 31) Cactus Grid Computational Toolkit <http://www.cactuscode.org>
- 32) The Globus Grid Project <http://www.globus.org>
- 33) APache Jetspeed Portal <http://jakarta.apache.org/jetspeed/site/index.html>
- 34) Common Component Architecture <http://www.cca-forum.org/>
- 35) JXTA Peer-to-Peer Environment <http://www.jxta.org>
- 36) G.C. Fox, D. Gannon and M. Thomas, 『Editorial: A Summary of Grid Computing environments』、Concurrency and Computation: Practice and Experience Vol. 14, Grid Computing environments Special Issue 13-15, 1035-1044 ページ、(2002)
- 37) Fran Berman, Geoffrey Fox and Tony Hey, 『Grid Computing: Making the Global Infrastructure a Reality』、ISBN 0-470-85319-0, John Wiley & Sons Ltd, Chichester, (2003), <http://www.grid2002.org> を参照。
- 38) Geoffrey Fox, Dennis Gannon, and Mary Thomas, 『Overview of Grid Computing environments』、参考資料 [37]の第 20 章, 543-554 ページ
- 39) Craig Lee and Domenico Talia, 『Grid Programming Models: Current Tools, Issues and

- Directions₁, 参考資料 [37]の第 21 章、555-578 ページ
- 40) Henri Casanova and Fran Berman, 『Parameter Sweeps on the Grid with APST₁』, 参考資料 [37]の第 33 章、773-788 ページ
 - 41) Hidemoto Nakada, Yoshio Tanaka, Satoshi Matsuoka and Staoshi Sekiguchi, 『Ninf-G: a GridRPC system on the Globus Toolkit₁』, 参考資料 [37]の第 25 章、625-638 ページ
 - 42) Gabrielle Allen, Tom Goodale, Michael Russell, Edward Seidel and John Shalf, 『Classifying and enabling Grid applications₁』, 参考資料 [37]の第 23 章、601-614 ページ
 - 43) Andrew S. Grimshaw, Anand Natrajan, Marty A. Humphrey, Michael J. Lewis, Anh Nguyen-Tuong, John F. Karpovich, Mark M. Morgan and Adam J. Ferrari, 『From Legion to Avaki: The Persistence of Vision₁』, 参考資料 [37]の第 10 章、265-298 ページ
 - 44) Gregor von Laszewski, Jarek Gawor, Sriram Krishnan and Keith Jackson, 『Commodity Grid Kits - Middleware for Building Grid Computing environments₁』, 参考資料 [37]の第 26 章、639-658 ページ
 - 45) Tomasz Haupt and Marlon E. Pierce, 『Distributed object-based grid computing environments₁』, 参考資料 [37]の第 30 章、713-728 ページ
 - 46) Sudesh Agrawal, Jack Dongarra, Keith Seymour, and Sathish Vadhiyar, 『NetSolve: Past, Present, and Future; A Look at a Grid Enabled Server₁』, 参考資料 [37]の第 24 章、615-624 ページ
 - 47) William A. Watson, Ying Chen, Jie Chen and Walt Akers, 『Storage Manager and File Transfer Web Services₁』, 参考資料 [37]の第 34 章、789-804 ページ
 - 48) V. Mann and M. Parashar, 『DISCOVER: A Computational Collaboratory for Interactive Grid Applications₁』, 参考資料 [37]の第 31 章、729-746 ページ
 - 49) Geoffrey Fox and Shrideep Pallickara, 『NaradaBrokering: An Event Based Infrastructure for Building Scaleable Durable Peer-to-Peer Grids₁』, 参考資料 [37]の第 22 章、579-600 ページ
 - 50) David Snelling, 『Unicore and the Open Grid Services Architecture₁』, 参考資料 [37]の第 29 章、701-712 ページ
 - 51) Mary Thomas and Jay Boisseau, 『Building Grid Computing Portals: The NPACI Grid Portal Toolkit₁』, 参考資料 [37]の第 28 章、675-700 ページ
 - 52) Rich Wolski, John Brevik, James S. Plank, and Todd Bryan, 『Grid resource allocation and control using computational economies₁』, 参考資料 [37]の第 28 章、747-772 ページ
 - 53) Dennis Gannon, Rachana Ananthkrishnan, Sriram Krishnan, Madhusudhan Govindaraju,

Lavanya Ramakrishnan, and Aleksander Slominski, 『Grid Web services and application factories』、参考資料 [37]の第9章、251-264 ページ

- 54) Concurrency and Computation : Practice and Experience
<http://www3.interscience.wiley.com/cgi-bin/issuetoc?ID=102522447>
- 55) グローバルグリッドフォーラムのGCE研究グループ、
http://www.gridforum.org/7_APM/GCE.htm.
- 56) Open Grid Services Architecture (OGSA)
http://www.gridforum.org/ogsiwg/drafts/ogsa_draft2.9_2002-06-22.pdf
- 57) GrADS (Grid Application Development Software Project)
<http://www.hipersoft.rice.edu/grads/>
- 58) AVS <http://www.avsc.com/>
- 59) Khoros <http://www.khoros.com/>
- 60) BPEL4WS Business Process Execution Language for Web Services
<http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/>
- 61) WSCL Web Services Conversation Language <http://www.w3.org/TR/wscl10/>
- 62) UK e-Science Program の ICENI プロジェクト (<http://www.lesc.ic.ac.uk/iceni/>)