社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

カスタマイズ可能な仮想計算機上におけるグリッドでのジョブ実行

山形　育平[†]　青木　孝文[†]　高宮　安仁[†]　中田　秀基[††,†]　松岡　聡[†,†††]

† 東京工業大学　〒 152–8552 東京都目黒区大岡山 2–12–1
†† 産業技術総合研究所　〒 305–8568 茨城県つくば市梅園 1–1–1
††† 国立情報学研究所　〒 101–8430 東京都千代田区一ツ橋 2–1–2
E-mail: †yamagat1@is.titech.ac.jp

あらまし　近年グリッドでは、ユーザーが投入するジョブが多様化しており、それに伴いユーザーの求める実行環境も多様化している。しかし、すべてのサイトでユーザーの求める実行環境が提供されているとは限らない。そこで我々は仮想計算機とユーザーがカスタマイズ可能なインストレーション機構を組み合わせ、ユーザーがジョブ実行環境を容易に構築できるシステムを提案する。本システムでは GRAM 経由で、カスタマイズ可能なインストレーション機構である Lucie [5] を呼び出し、これを用いて仮想計算機の構築を行い、その計算機上でジョブを実行することができる。このシステムを評価した結果、既存の計算機環境に影響を与えないジョブの実行が可能であった。環境構築時間も一般的なグリッドジョブの実行時間と比べて許容範囲であることを確認し、本研究の有効性を確認した。
キーワード　グリッド、仮想計算機

Job execution in Grid on customizable virtual machine

Ikuhei YAMAGATA[†], Takafumi AOKI[†], Yasuhito TAKAMIYA[†], Hidemoto NAKADA[††,†], and Satoshi MATSUOKA[†,†††]

† Tokyo Institute of Technology　Oookayama 2–12–1, Meguro-ku, Tokyo, 152–8552 Japan
†† National Institute of Advanced Industrial Science and Technology　Umezono 1–1–1, Tsukuba, Ibaraki, 305–8568 Japan
††† National Institute of Informatics　Hitotsubashi 2–1–1, Chiyoda-ku, Tokyo, 101–8430 Japan
E-mail: †yamagat1@is.titech.ac.jp

Abstract　Grid applications in the Grid environment become diverse recently. It makes execution environments which user requires diverse. However, execution environments which user requires are not always provided in the heterogeneous grid environment. So we propose a system which can construct an execution environment which user requires on any machine. It couples a virtual machine and customizable installation framework. The result of an experiement shows a validity of our system.
Key words　Grid computing, virtual machine

1. はじめに

近年、計算機環境としてグリッドが広く用いられている。グリッドとは、広域的に分散した計算機資源をユーザーの需要と資源提供者のポリシーを元に、ネットワークを通じて動的に共有し、協調動作させるものである。そこではユーザーは多種多様なジョブを実行しようとする。また、ジョブの多様化に伴い、ジョブが実行環境として要求するオペレーティングシステムやアプリケーション、ソフトウェアライブラリなども多様化している。一方でグリッドのように複数の計算機資源を使用する環境では、各々の計算機資源が管理者のポリシーによってオペレーティングシステムやアプリケーション、ソフトウェアライブラリなどがインストールされている。そのためユーザーが望む実行環境が構築されているとは限らない。

このような場合においてジョブを実行するためには、実行前に計算機上にユーザーが必要とする実行環境をセットアップする必要がある。しかし、ジョブの実行ごとに環境変更を行うことは他のユーザーへの影響が大きく、管理ポリシーによっては不可能な場合が多い。そのため、結果としてジョブを実行できないということがある。このためユーザーが望む任意のジョブ

実行環境を、管理ポリシーや既存の環境に関係なく、動的に構築できる技術が求められている。

これを解決する手段として、仮想計算機を用いてグリッドを構築する方法がある。仮想計算機とは、ソフトウェアによって、物理的な計算機の上に仮想的な計算機を構築する技術のことである。仮想計算機はユーザー権限で起動可能であり、そのためその設定はユーザーが自由にカスタマイズできる。また、仮想計算機上でどのようなジョブが実行されていても、実計算機やその他の仮想計算機に影響を与えないため、安全にジョブを実行できる。さらに、同じディスクイメージを共有することで、すべてのノードで同じジョブ実行環境を構築することができる。よって仮想計算機上にユーザーの望む実行環境を容易に構築できる技術が求められている。

そこで本研究では、この仮想計算機とカスタマイズ可能なインストレーション機能を組み合わせることにより、ユーザーが容易にカスタマイズ可能なジョブ実行環境の構築を可能にするシステムを設計、実装した。また本システムを用いて Blast 実行環境の構築とジョブの実行を行い、既存の計算機環境に影響を与えない、ジョブの実行が可能であることを確認した。またこのシステムでの 1 ノードの構築時間は 199 秒で、通常グリッドで実行されるアプリケーションが長時間実行されるものが多いことから考えると、十分に許容範囲内であるといえる。

2. グリッドに求められる要件とその解決手法

グリッドでは、複数の分散した計算機資源を、ネットワークを通じて動的に共有させる。そのため以下のような特徴が存在する。

- アーキテクチャ、管理体制の不均質性

複数の計算機資源を利用するため、各計算機間ではアーキテクチャやインストールされているオペレーティングシステムやアプリケーションなど様々な点で異なっている可能性が高い。

- ユーザーの権限が限定的

ユーザー自身がグリッド上の計算機環境をカスタマイズしたり、再構築することはできない。

グリッドではこのような環境にもかかわらず、多様なジョブを実行することが求められている。そのため我々は以下の 2 つの要件がグリッドに必要であると提案する。

- ユーザーがカスタマイズ可能なジョブ実行環境

ユーザーがジョブ実行環境をカスタマイズすることを可能にすることにより、ユーザーにとっては自分の望む実行環境を構築でき、また管理者にとってはユーザーのために計算機環境を変更する手間が無くなる。

- 資源とユーザープロセスの安全な分離

資源とユーザープロセスを分離しなければ、ユーザープロセスにより計算機資源を破壊される可能性があり、管理者や他のジョブ実行ユーザーにも影響を与えてしまう。したがって資源とユーザーを完全に分離する必要がある。

我々は以上の要件を満たすシステムを提案する。これは仮想計算機とカスタマイズ可能なインストレーション機構を用いることによって実現可能である。

2.1 仮想計算機

仮想計算機とは、ソフトウェアによってハードウェアを仮想化し、物理的な計算機の上に仮想的な計算機を構築することで、独自の環境（オペレーションシステムやアプリケーションソフト）を持った複数の仮想計算機を同時に実行することができるものである。各サービスを各仮想計算機上で一つずつ実行することにより、保守性をあげる一方、それに伴うスペースや電力の効率化の恩恵を得ることができる。また、ハードウェアを仮想化していることで、CPU アーキテクチャなどを考慮せず、移植性の高いソフトウェアを生み出すのにも適しているといった利点がある。

仮想計算機上で動くオペレーティングシステムは ”ゲストオペレーティングシステム（ゲスト OS）”、仮想計算機自体を動作させているオペレーティングシステムは ”ホストオペレーティングシステム（ホスト OS）”と呼ばれる。ゲスト OS には、ホスト OS とは異なるオペレーティングシステムを動作させることが可能である。

また、仮想計算機を使用することにより、管理者の計算機資源を安全に守ることも可能となる。仮にゲスト OS の中でプログラムが実行環境を破壊されたとしても、ホスト OS には影響を与えない。仮想計算機のメモリやディスク容量は起動時に設定することが可能であり、メモリやディスクが必要以上に使用されるのを避けることが可能である。

仮想計算機技術として VMWare [8] [9] [14] 、Xen [12] [17] 、coLinux [1] [16] 、UML(User Mode Linux) [11] [15] などがあり、それぞれ性能面においてトレードオフが存在する。より性能のいい実行環境を提供するために、ユーザーの要求に合わせて選択可能にすることが必要である。

2.2 カスタマイズ可能なインストレーション機構

カスタマイズ可能なインストレーション機構とは、ユーザーがオペレーティングシステム、カーネル、ソフトウェア、ライブラリ、環境設定などを自由に選び、そのような環境のマシンを容易にインストール可能にするシステムである。

仮想計算機の OS をインストールする際にこの機構を使用すると、ユーザーが任意に指定した環境を容易に構築することができる。加えて、このシステムに選択できるソフトウェアを制限する機構を付け加えることにより、セキュリティホールのあるプログラムをインストールできなくしたり、管理者ポリシーに沿ったジョブ実行環境を構築することが可能になる。また、ユーザーに実際にインストール作業を行わせないことにより、余計なアプリケーションをインストールされることを防ぐことができる。このような技術として Lucie [5]、RedHat KickStart [18]、NPACI Rocks [10] などがある。

3. 本研究の提案するシステムの構成

2.1 と 2.2 で紹介したシステムを用いた、我々が提案するシステムの構成は次のようである (図 1 参照)。

（1） ユーザーはジョブ実行サービスに対し、実行するジョブおよびジョブ実行環境設定ファイルを転送する。ジョブ実行環境設定ファイルには構築するノード数、メモリ、HDD、kernel

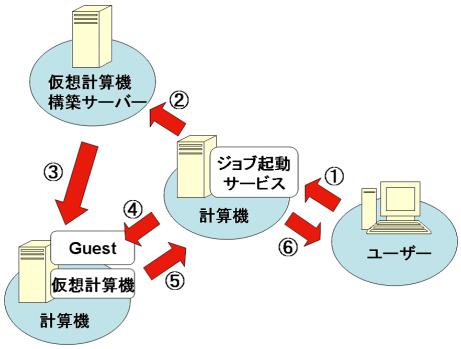

図 1　提案の概要
Fig. 1　abstract of design

のバージョン、インストールするアプリケーションなどを記述する。

このときユーザーごとに、構築できるノード数、使用できるメモリや HDD などを制限することができる。これは特定のユーザーが計算機資源を無駄に消費させないためにも必要である。

（2） ジョブ起動サービスはユーザーから送られてきたジョブ実行環境設定ファイルを仮想計算機構築サーバに送る。

（3） 仮想計算機構築サーバーによって仮想計算機上にジョブ実行環境を構築する。

（4） ジョブ起動サービスはユーザーから送られてきたジョブを仮想計算機に送る。

（5） 仮想計算機は送られてきたジョブを実行し、結果をジョブ起動サービスに送る。

（6） ジョブ起動サービスは実行結果をユーザに送る。

これにより、どのような環境下においても、ユーザーは自分が望む環境を構築し、そこでジョブを走らせることができる。

4. プロトタイプ実装

我々は 3. 章で提案したシステムをもとに、カスタマイズ可能な計算機環境構築を行うシステムのプロトタイプ実装を行った。本章ではこのシステムを説明する。

ジョブ起動サービスとして Globus [2] の GRAM(Grid Resource Allocation Manager) を、仮想計算機構築サーバーとして Lucie [5] を、仮想計算機として VMWare [14] を使用した。

4.1 GRAM

GRAM はグリッド上で標準的に用いられている Globus Toolkit のジョブ起動機構である。GRAM の特徴として jobmanager モジュールの追加が可能であるということがある。そこで我々は指定されたジョブを fork(2) して実行する jobmanager-fork モジュールを元に、仮想計算機の起動や lucie システムのセットアップなどを行うシステムを追加し、我々の提案したシステムが起動するあたらしい jobmanager モジュールである jobmanager-vm モジュールを作成した。

4.2 Lucie

Lucie は大規模なクラスタ環境を容易に高速に構成することを目的として作られた、高速セットアップ・管理ツールである。新しいクラスタの構築やクラスタの再構成の際に使用することにより、1 台ごとに手作業でインストールする場合と比べてセットアップ時間を短時間にし、かつ確実に行うことができる。

Lucie ではインストールしたいアプリケーションを柔軟に選択することが可能であり、インストールするカーネルのバージョンや HDD のパーティションなど様々なものが設定可能である。また各種アプリケーションの設定ファイルは、各マシンごとに異なるものを配置することが可能である。また pxe ブートによりインストールが始まるため、マシンの起動または再起動を行うだけで新しい環境が自動的に構築される。

実際に Lucie を用いてインストールを行う場合、計算機管理者はまず Lucie の設定ファイルを書く。これにはインストールを行うマシンの情報、インストールするパッケージやカーネル、パッケージサーバーのアドレスなど、様々な情報を記述する。次に Lucie システムのセットアップを行う。ここでは設定ファイルに基づいてインストールイメージを作り、マシンのインストールに必要なシステムの設定を行う。この後マシンを起動、再起動するとインストールが行われるようになる。

4.3 実 行 手 順

現在の実装を詳しく説明する (図 2 参照)。今回は簡単のために実計算機”nodeA”上に、仮想計算機”VMnodeB”を立てる場合を説明する。

（1） 環境設定ファイルの作成

ユーザーは memory、HDD、カーネルのバージョン、必要なパッケージを指定する。このとき GUI を通してファイルを生成する。ユーザーは GUI の画面を見ながら、対話的にジョブ実行環境設定ファイルの作成が行える。

（2） 設定ファイルの転送

ユーザーは GRAM サーバーに対し、実行するジョブおよびジョブ実行環境設定ファイルを転送する。

（3） Lucie と VMWare の設定ファイルの作成

GRAM はユーザーから送られてきたジョブ実行環境設定ファイルを元に、Lucie の設定ファイルと VMWare の設定ファイルを作成する。

（4） 仮想計算機を立ち上げるマシンの選択と仮想計算機のアドレスの決定

GRAM は仮想計算機を立ち上げるマシンと、そこで立ち上げる仮想計算機の MAC アドレス、IP アドレスを決める。実際には立ち上げるマシンやアドレスは固定となっている。

（5） Lucie システムのセットアップ

仮想計算機構築を行う Lucie は、Lucie の設定ファイルを元に、インストレーション機能のセットアップを行う。

（6） VMWare の設定ファイルの転送

GRAM は VMWare の設定ファイルを、仮想計算機を立ち上げる実計算機 nodeA に転送する。

（7） 仮想計算機の構築

実計算機 nodeA は VMWare の設定ファイルを元に、指定された容量の仮想ディスクを作成、その後仮想計算機 VMnodeB を構築する。

（8） ゲスト OS インストール

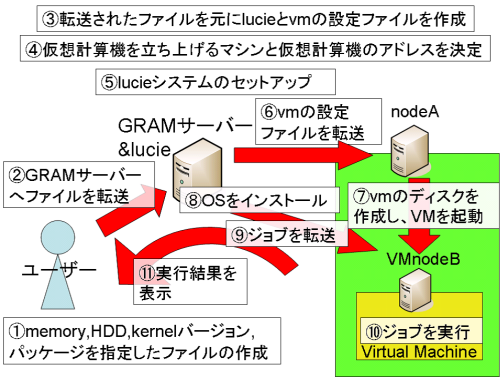

図 2　実装したシステム図

Fig. 2　Implementation system

表 1　評 価 環 境

Table 1　evaluation environment

CPU	AMD Opteron(tm) Processor250(2.4GHz) × 2
メモリ	PC2700 2GB
OS	Debian GNU/Linux sarge
カーネル	2.4.27
ネットワーク	Gigabit Ethernet
VMWare	5.0(linux 版)

Lucie によって仮想計算機 VMnodeB にユーザーが指定したカーネルやアプリケーションがインストールされ、ジョブ実行環境が構築される

（9） ジョブの転送

GRAM はユーザーから送られてきたジョブを仮想計算機 VMnodeB に送る。

（10） ジョブの実行

仮想計算機 VMnodeB は送られてきたジョブを実行する。

（11） ジョブ結果の出力

仮想計算機 VMnodeB は GRAM サーバー経由で実行結果をユーザーに送る。

現在の実装ではあるひとつのノードがジョブ起動サービスを行うマシンと仮想計算機構築サーバーを兼ねる仕様になっている。またすべてのシステムは Debian GNU/Linux [19] で動いており、インストールできるディストリビューションも Debian の woody に限られている。

5. 評　　価

本章では、実装を行ったシステムの評価実験、考察を行う。

5.1 評 価 環 境

本実験の評価環境として、松岡研究室 PrestoIII クラスタを使用した。スペックは表 1 の通りである。また構築する仮想計算機のスペックは表 2 のとおりである。

本システムの有効性を評価するために、本稿では Blast(Basic Local Alignment Search Tool) [4] が実行可能な環境の構築を行った。BLAST は相同性検索ソフトウェアで、データベースとよばれるファイルに対し、特定の配列の相同性検索を行うものである。Blast を実行するために必要なパッケージであるが、

表 2　仮想計算機のスペック

Table 2　spec of virtual machine

HDD	20GB(SCSI buslogic)
メモリ	256MB
OS	Debian GNU/Linux woody
カーネル	2.4.28
インストールする追加パッケージ	blast2

表 3　仮想計算機構築時間

Table 3　time of virtual machine(1)

ファイル転送と認証	lucie システムのセットアップ	VM 起動と OSinstall	再起動	総構築時間
42 秒	64 秒	59 秒	34 秒	199 秒

表 4　ゲスト OS インストール時間の詳細

Table 4　time of install guestOS(s)

partition 設定 &format	package の取得	package のインストール	kernel のインストール	設定ファイルの更新
9.25 秒	13.96 秒	25.93 秒	8.58 秒	1.78 秒

debian では BLAST2 というパッケージが用意されている。そこで、ジョブ実行環境設定時に、必要な追加パッケージとして blast2 を選択した。また、実際に動かすジョブとしては、データベースファイルに対する、配列の相同性検索を行った。

5.2 実行環境構築時間の評価

まずは仮想計算機を 1 台構築し、それにかかった時間の評価、考察を行う。環境構築時間は表 3 のようになり、仮想計算機構築に全体で 199 秒要した。通常 BLAST のジョブ実行時間は数時間以上であり、それと比較すると、この構築時間は十分許容範囲であると考えられる。また、実際にグリッド上の場合を考えると、グリッドのアプリケーションは長時間実行をするものが多いため、本システムは十分適用可能だと言える。

一方で、環境構築時間を短縮することは重要である。ファイル転送&認証、再起動時間の削減は難しく、可能なのは lucie システムのセットアップとゲスト OS のインストール時間である。ここでは後者にかかっているの時間の削減方法を考える。

ゲスト OS のインストールには 59 秒かかっており、これは全体の 30%を占めていることになる。この内訳を調べると、表 4 のようになった。パッケージの取得に 13.96 秒、パッケージのインストールに 25.93 秒、kernel のインストールに 8.58 秒、合計で 48.47 秒かかっており、インストール時間の多くを占めることがわかる。これを削減する方法として、一度構築したジョブ実行環境のイメージを保存しておき、それを再利用する方法がある。これによりゲスト OS を立ち上げる際に行うことが、HDD のパーティション設定とフォーマット、アプリケーションの設定ファイルの更新だけになり、環境構築時間の大幅な削減につながる。このようなイメージを保存し、再利用するシステムを構築することが今後の課題のひとつである。

5.3 複数実行環境時間の評価

次に複数台同時に構築を行い、その構築時間を計測した。本

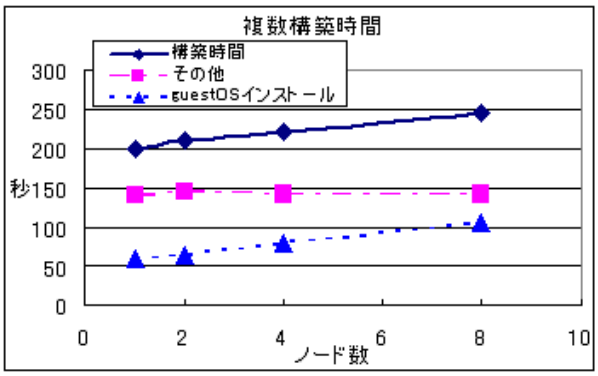

図 3　複数仮想計算機構築時間
Fig. 3　time of virtual machines

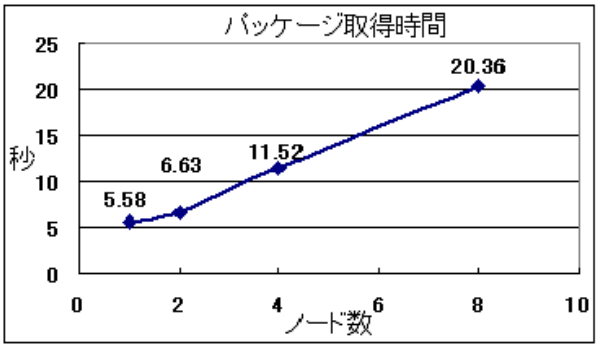

図 4　パッケージ取得時間の推移
Fig. 4　time of getting package

稿では 8 台までの複数台セットアップにおけるオーバヘッドを計測した (図 3)。ノードが増加するにつれて構築時間も増加しており、それがゲスト OS のインストール時間に由来しているものであることがわかる。これはゲスト OS 構築のときのパッケージ取得のために、パッケージサーバーに負荷が集中していることが原因である。実際にパッケージ取得時間の推移を図 4 に示す。ノードが増加するにつれ、パッケージ取得時間も増加していることがわかる。

このことは計算機の台数が少ないときはあまり問題にならないが、仮想計算機の構築が数十台、数百台規模になったときには大きな性能低下が予想される。このためパッケージサーバーのレプリケーションや Dolly+ [13] などの高速ファイル転送システムを使用することが必要である。

6. 関 連 研 究

仮想計算機を用いて、ジョブの実行環境を構築する研究はいくつか存在する。

VMPlant [6] はユーザーのジョブ実行環境として、仮想計算機を提供するシステムである。どのような仮想計算機を構築するかは DAG(有向非巡回グラフ) を用いて設定する。仮想計算機を構築する際に使用できる仮想計算機技術として、VMWare もしくは User-mode Linux が選択できる。また、よく使われる仮想計算機のディスクイメージをゴールデン VM として保存し、これを再利用することにより仮想計算機の起動の高速化をはかっている。このゴールデン VM を必要な数のホスト上に展開することで、均一な仮想計算機環境を容易に構築できる。

VMPlant ではゴールデン VM を利用したインクリメンタルな仮想計算機の構築が可能であり、仮に構築したい環境とゴールデン VM が非常に近い場合、高速に環境構築が可能である。一方で、VMPlant ではインストールするアプリケーションなどを DAG を用いて記述しなければならず、自分が構築したい環境を作る手続きが複雑である。我々のシステムではインストールするアプリケーションを GUI 画面を用いて選択できるように、より柔軟に環境をカスタマイズすることができる。

また VMPlant では仮想計算機の IP アドレスや MAC アドレスをユーザーが指定しなければならない。そのためユーザーが実計算機の環境を熟知していなければならない。またユーザーが IP アドレスや MAC アドレスを指定することにより、アドレス衝突の恐れがあり、セキュリティ的にも問題がある。我々のシステムでは、仮想計算機の IP アドレスや MAC アドレスをユーザーが決める仕様になっておらず、そのためユーザーは計算機の環境を知らなくてもジョブを実行することができる。

The Virtuoso Model [7] は仮想計算機グリッドのための仮想ネットワークの研究である。資源提供者の仮想計算機を仮想ネットワークによってローカルネットワークに接続し、ユーザーはあたかも目の前にある計算機のように使えるようにすることが可能になる。ユーザーは仮想計算機のハードウェアやメモリ、HDD などの基本的な性能は指定することができる。

一方で、基本的なオペレーティングシステムとアプリケーションは最初から指定されており、使用したいアプリケーションがある場合は自らがインストールする必要がある。そのため、ユーザーへの負担は大きい。特に大規模な仮想計算機群を構築しようとする際には非常に大きいコストとなる。我々のシステムでは仮想計算機にインストールするアプリケーションを選択肢で選ばせることにより、ユーザーにとっても自分の構築したい環境を用意に選ぶことができ、計算機管理者にとっても構築する環境を制限することが可能である。

7. お わ り に

本研究では、複数の計算機資源を同時に使用する環境で、ユーザの望むジョブ実行環境を実現するために、カスタマイズ可能なインストレーション機能と仮想計算機を使った、既存の計算機環境に影響を与えない、カスタマイズ可能なジョブ実行環境を提案、設計および実装を行った。そして Blast を実行するために必要な環境を構築し、実際にジョブが動作すること、複数台のジョブ実行環境を構築でき、その構築にかかる時間も一般的なグリッドで実行されるジョブの実行時間と比較すると十分許容範囲であることを確認し、本システムのジョブ実行環境としての有効性を確認した。

今後の課題として、次のことがあげられる。

- ジョブ実行環境構築時間の短縮：仮想計算機イメージの再利用

一度構築した仮想計算機のディスクイメージを保存しこれを再利用することにより、ゲスト OS のインストール時間の大部

分を削減することが可能である。

- 複数の仮想計算機の選択

現段階では仮想計算機として VMWare しか選べない。Xen、UML など仮想計算機技術にはさまざまあり、ユーザーの要求にあわせて選択可能にすることが必要である。

- 仮想計算機を立ち上げる実計算機の選択機構の構築

現在の実装では、仮想計算機を立ち上げるマシンは固定となっており、これを動的に決める機構がない。本来ならば仮想計算機を立ち上げる候補のマシンの CPU 使用率や memory の使用量、HDD の容量や仮想計算機がいくつ立ち上がっているかなどを元に、動的に柔軟に判断することが必要だと考える。今後適切なノードを動的に選ぶシステムをつくることが必要である。

- ジョブの監視機構の構築

現在の実装では、仮想計算機で動かすジョブを指定させてはいるが、これを管理、監視する機構が存在しない。通常グリッドではどのようなユーザーがどのようなジョブが実行しているかを管理できていてしかるべきである。サイト管理者のポリシーに従い、ジョブの監視、場合によっては拒否するようなシステムが必要である。

- よりセキュアなジョブ実行環境の構築

現在の仮想計算機を用いたジョブ実行環境では、CPU リソースやネットワークリソースを制限することはできない。他のユーザやサイト管理者に影響を与えないようにするためにも、これらのリソースに対し制限をかけることが必要である。

- 計算機資源の管理

現段階ではユーザーが求める計算機資源の要求に対して、サイト管理者はこれに制限や拒否をすることはできない。しかし実際の Grid では、ユーザーのリソース使用に対して、なんらかの制限をかける必要があり、このようなシステムを作ることが必要である。また GWiQ-P [3] のような、ユーザーが使用しているリソースをグリッド全体で監視し、制限するシステムも必要である。

謝辞 本研究の一部は，独立行政法人新エネルギー・産業技術開発機構 基盤技術研究促進事業 (民間基盤技術研究支援制度) の一環として委託を受け実施している「大規模・高信頼サーバの研究」の成果である.

文　献

[1] Dan Aloni. Cooperative linux. Proceedings of the Linux Symposium, volume 1, pp.p23-p32, 2004

[2] Ian Foster and Carl Kesselman. Globus: A metacomputing infrastructure toolkit.: The international Journal of Supercomputer Applications and High Performance Computing, pp. p115-128, 1997.

[3] Kfir Karmon, Liran Liss, Assaf Schuster. GWiQ-P: An Efficient Decentralized Grid-Wide Quota Enforcement Protocol. The 14th IEEE International Symposium on High Performance Distributed Computing(HPDC-14), 2005

[4] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers and David J. Lipman. Basic Local Alignment Search Tool. J. Mol. Biol. 215:403-410, 1990.

[5] 高宮安仁, 真鍋篤, 松岡聡.: Lucie:大規模クラスタに適した高速セットアップ・管理ツール.: Symposium on Advanced Computing Systems and Infrastructures, May 2003.

[6] Ivan Krsul, Arijit Ganguly, Jian Zhang, Jose A.B Fortes, Renato J. Figueiredo: VMPlants: Providing and Managing Virtual Machine Execution Environments for Grid Computing: Proceedings of the 2004 ACM/IEEE conference on Supercomputing, page7, 2004

[7] Ananth I. Sundararaji and Peter A. Dinda. Toward virtual networks for virtual machine grid computing. Virtual Machine Research and Technology Symposium, pp. 177-190, 2004.

[8] Jeremy Sugerman, Ganesh Venkitachalam, and Beng-Hong Lim. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor. Proc. USENIX Ann. Technical conf., June 2001

[9] Carl A. Waldspurger. Memory Resource Management in VMware ESX Server. Proc. 5th Symposium on Operating Systems Design and Implementation(OSDI'02), Dec. 2002

[10] Mason J.Katz, Philip M. Papadopoulos, and Greg Bruno. Leveraging standard core technologies to programmatically build linux cluster appliances. CLUSTER 2002, IEEE International Conference on Cluster Computing, April 2002.

[11] Jeff Dike. A User-mode Port of the Linux Kernel. Proc. 4th Ann. Linux Showcase and Conf., USENIX Association, Atlanta, GA, Oct. 2000

[12] Paul Barham, Boris Dragovic, Keir Fraser, Steven.Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warifield. Xen and the Art of Virtualization. Proc. ACM Symp. Operating Systems Principles (SOSP 2003), Oct. 2003.

[13] Atsushi Manabe. Disk cloningprogram 'dolly+' for system management of pc linux cluster. Computing in High Energy Pysics and Nuclear Physics, 2001.

[14] vmware web site
http://www.vmware.com/

[15] UML: User-mode Linux web site
http://user-mode-linux.sourceforge.net/

[16] colinux web site
http://www.colinux.org

[17] xen web site
http://www.cl.cam.ac.uk/Research/SRG/netos/xen

[18] Redhat linux kickstart information.
http://wwwcache.ja.net/dev/kickstart/.

[19] Debian GNU/Linux
http://www.debian.org/