2007－DPS－133 (16)
2007／11／27

仮想マシンにおけるLinux Gridの考察

田中堅一　　上原 稔　　森 秀樹
東洋大学大学院 工学研究科 情報システム専攻

近年では科学技術計算分野やビジネス分野、さらには映像技術の分野でもコンピューティング需要が高まりつつあり、遊休資源を活用するグリッドコンピューティングを利用することが検討されている。グリッドは高効率計算（HTC）を主眼に置いた分散コンピューティング技術であり、またオープンソースソフトウェアによって提供されるミドルウェアも存在し、クラスタなどと比較して低コストで構築できる利点がある。しかし現状のグリッドでは最も普及しているWindows OSを搭載したPCを十分に活用することは不可能であり、仮想マシンを使用しなければならない場合もある。本論文では、Windows上に実現されたグリッドと仮想マシンにおけるLinuxグリッドの比較を行う。

A Case Study on Linux Grid using Virtual Machine

Kenichi Tanaka　Minoru Uehara　Hideki Mori
Toyo University Graduate School, Department of Open Information Systems

Recently, the computing demands for the science and technology calculation, the business, and the imaging technique are rising. The grid computing that uses the idle resource is discussed as the platform for such applications. Grid is the distributed computing technology that put High-Throughput Computing (HTC) in the principal objective. Moreover, several middleware are offered as open source software, and then Grid has an advantage that it can be constructed with cost lower than the Cluster. However, in current grids, it might have to be impossible to use PC equipped with Windows OS. Therefore, we may have to use a virtual machine. In this paper, we compare the Linux grid in the virtual machine with Windows grid.

1. はじめに

近年では科学技術計算の分野やビジネスの世界においてもコンピューティング需要が増加している。グリッドコンピューティングというものは、ネットワークを通じて遠隔地にあるPCを相互接続し、それらの計算資源を利用するものである。また、グリッドはオフィスや大学などに配置されたPC群が使用されていない場合に、それらの計算資源を利用するものであり、「遊休CPUの活用」という利用も可能である。

グリッドに関する研究はこれまでにも数多く行われてきている。日本国内においてもNational Research Grid Initiative (NAREGI)プロジェクトが存在する（文献[1]）。

さらに今日では商用製品も販売されるようになるなど、グリッド技術は成熟してきたように感じられる。しかしながらグリッドを利用可能なプラットフォームはUNIX / Linux系のシステムに限られている場合がほとんどであり、Windowsを対象としたグリッドはそのほとんどが、すべての機能を使えない状態にある。たとえば富士通の商用グリッド製品であるSystemwalker CyberGRIPでは、計算ノードとしてのみWindowsマシンを利用することができる程度である。だが、Windows OSは現状で最も普及しているOSであり、Windowsマシンをグリッドに利用することが可能であれば、膨大な計算資源を確保することが可能である。

また、グリッド環境を構築する際には莫大なコストをかける必要がある。前述のCyberGRIPでは、マスターサーバの導入に百万円単位の導入コストをかけることになる。このような問題に対しては、オープンソースによって提供されるミドルウェアを用いることでコストの大幅な削減を期待することができる。オープンソースのミドルウェアとして、Globus ToolkitやBOINCなどがある。

映像技術の世界においても、より高精細な画像を製作するために、グリッドを利用してレンダリングを行うということが試みられている。このような背景から本研究では、評価用のアプリケーションとして3DCGのレンダリングを行う。

我々は文献[2]において、オープンソースソフトウェアを用いてWindowsにおけるグリッドが構築可能であることを示した。また、マルチコアプロセッサを有効に活用することで、1台のマシンを複数のノードとして扱い、グリッドシステム全体の性能を向上させることが可能であることを文献[3]で示した。

だがWindowsにおいてグリッドを構築する方法として、仮想マシンを利用する方法もある。文献[2]においては直接Windowsグリッドを構築したが、仮想マシンを用いて、Linuxグリッドを構築することでもWindowsマシンの計算

資源を利用することは可能である。そこで本研究では、仮想マシン上にオープンソースソフトウェアを用いた Linux グリッドを構築し、Windows におけるグリッドとの比較考察を行う。

本論文の章構成は次の通りである。関連する技術については 2 章で解説を行う。またシステムの概要については 3 章で示す。そして評価及び考察について 4 章、5 章にて論ずる。総合的なまとめを 6 章において行う。

2. 関連技術

2.1 Grid ミドルウェア

ミドルウェアとは、OS とアプリケーションとの間に入り、アプリケーションがおこなう処理の中で共通で普遍的な処理を行うソフトウェアである。Grid においては、グリッドシステムを構築するための基盤となるソフトウェアを指す。オープンソースの Grid ミドルウェアとして、BOINC (Berkeley Open Infrastructure for Network Computing, 文献[4]) や Globus Toolkit (文献[5]) などが存在する。

またミドルウェアはグリッドプロジェクト毎に開発されることも多く、NAREGI プロジェクトにおいても専用のミドルウェアが用意されている。

2.1.1 BOINC

BOINC は地球外生命体の探索を行う分散コンピューティングプロジェクトである SETI@home (文献[6]) を支援する目的で開発された、マルチプラットフォーム対応の分散コンピューティング環境である。また文献[4]によれば、BOINC は「ボランティア・コンピューティングとデスクトップ・グリッド・コンピューティングのためのオープンソースソフトウェア」という位置づけである。

BOINC クライアントは共通機能であるコア・クライアントと、プロジェクト毎に用意されるアプリケーション部分に分けられる。また、BOINC の特徴としてクライアントプログラムは、アプリケーション部分も含めて C 言語で記述される。そのため多量の計算を要するプロジェクト向きである。

しかしプロジェクトに必要なアプリケーションが C 言語で記述されることから、プラットフォーム依存となるアプリケーションが必要となる場合がある。すなわち、Linux 用のアプリケーションしか提供されない場合、たとえば Windows といった別のプラットフォームからはそのプロジェクトに参加することができない。

2.1.2 Globus Toolkit

Globus Toolkit は The Globus Alliance によって開発されている、オープンソースの Grid ミドルウェアである。本来はグリッドコンピューティングの研究用としてシカゴ大学の Ian Foster 教授によってはじめられたプロジェクトである。グリッドコンピューティングの標準化団体である Global Grid Forum (GGF, 現在は Open Grid Forum, 文献[7]) においては、グリッドの標準実装として採用された実績がある。また欧州では EGEE (Enabling Grids for E-sciencE, 文献[8]) プロジェクトにおけるグリッドミドルウェア gLite のベースとして採用されている。

Globus Toolkit の持つ特徴は、SOAP を中心としたプロトコルを採用している点である。SOAP は Globus Toolkit version 3 から使用されるようになったプロトコルである。Globus Toolkit 3 では、GGF により策定された Open Grid Services Architecture (OGSA) およびその仕様の規定である Open Gris Services Infrastructure (OGSI) を実装し、グリッドをサービスという形で実装する方式となった。OGSA は Web サービスをベースとしたアーキテクチャであり、Web サービスにサービスの状態管理機能を拡張したアーキテクチャである。

しかし、OGSA / OGSI は Web サービスをベースとしながらも、Web サービスとは別の技術である。そこで、The Globus Alliance はグリッドを Web サービスと同様に扱うために WS-Resource Framework を策定し、WSRF 準拠のグリッドミドルウェアとして、Globus Toolkit 4 を提供している。

Globus Toolkit は上記のように、グリッドを Web サービスとして実装することで、プラットフォームへの依存が少ない。また、コア機能は Java で実装されておりよりプラットフォームに依存しないようになっている。それでもまだ完全にマルチプラットフォームにまでは至っていない。特に Windows 上ではフルセットの Globus Toolkit を利用することは現状では不可能である。

なお、Globus Toolkit の商用利用を推進するための団体として、Globus Consortium (文献[9]) が HP、IBM などによって設立されている。さらに Globus Toolkit を商用製品として開発している企業に Univa (文献[10]) がある。Univa は IBM が同社製サーバで Globus Toolkit を利用するために提携した企業である。もともとはグリッドコンピューティング推進者らの手により、商用版 Globus Toolkit の開発とサポートを目的として設立された企業である。

2.1.3 NAREGI

NAREGI はナノテクノロジーなどの先端科学技術を主とした、サイエンスグリッド環境を実現することを目標に、グリッドミドルウェアの研究および開発を行うプロジェクトである。NAREGI グリッドミドルウェアは、ネットワークを通じてスーパーコン

ピュータやハイエンドサーバなどを接続し、OGSAに合ったサイエンスグリッドの構築を行うものとして開発されている。

なお NAREGI グリッドミドルウェアは現在、Linux、AIX、Solaris 環境で利用することが可能である。

2.2 仮想マシン

仮想マシン（Virtual Machine）とは、1台のコンピュータを複数のコンピュータとして扱うことができる技術である。具体的には、あるコンピュータ上に別のハードウェア環境を構築することで、別のコンピュータ環境を動作させる。またCPUエミュレータと呼ばれるQEMUも仮想マシンの一種である。QEMU は高い汎用性の反面、実行性能が十分でないため、グリッドには向かない。

仮想マシンは仮想ハードウェアの実現方法によって、ホストOS型と仮想マシンモニタ型に分けることができる（文献[11][12]）。

2.2.1 ホストOS型

ホスト OS 型の仮想マシンとは、ホスト OS 上でハードウェアをエミュレートし、その上でゲスト OS を動作させる方式の仮想マシンである。ハードウェアをソフトウェアでエミュレーションするため、実行速度は実マシンよりも遅くなる。しかし、既存の OS をそのまま利用できる利点がある。この方式の仮想マシンとしては、VMware Workstation（文献[13]）や Microsoft Virtual PC（文献[14]）等がある。

また同じホスト OS 型でも実現方法の違いにより処理性能に差が出る場合もある。Microsoft Virtual PC では実マシンの CPU と仮想マシンの CPU の速度比は 1:5 となる。またビデオチップもエミュレーションしているため、高度なグラフィックス機能を要求するソフトウェアを使用することは困難である。

対して VMware では、CPU エミュレーション時にカーネルモード命令のみをエミュレートする。ユーザーモードの命令は直接プロセッサによって実行される。これによりコード変換によるオーバーヘッドが少なくて済み、実ハードウェアに近い性能持つ。

2.2.2 仮想マシンモニタ型

この方式はハードウェア上で直接ハードウェアをエミュレートすることで、実ハードウェアとゲスト OS の間に仮想的なハードウェアを挟み込む。ホスト OS を介さないため、オーバーヘッドが少ないことが特徴である。この方式を採用している仮想マシンとしては、VMware ESX Server や Xen（文献[15]）などである。

Xen はオープンソースの仮想マシンモニタであり、仮想マシン環境に都合のよい仮想ハードウェアを再定義する準仮想化、実ハードウェアを実現する完全仮想化の二つの仮想化モデルを採用している。前者を利用する場合では OS の移植が必要となるが、オーバーヘッドが最小で済みパフォーマンスはほとんど低下しない。

なお、本研究の目的は Windows PC の計算資源を有効利用するためである。現状では Xen の上で Windows を利用した例はあるが、Windows ベースで Xen を利用したケースは見当たらなかった。そのため、本研究では仮想マシンモニタ型ではなく、ホスト OS 型の仮想マシンである VMware を使用した。

2.3 3DCG

3 次元コンピュータグラフィックス（3DCG）は、フォトリアリスティックな画像を製作するために利用される技術である。近年では映画やアニメーションの世界でも多用されており、中には全編 CG で制作された映画も存在する。特に 3DCG では、仮想 3 次元モデルから 2 次元画像を生成するレンダリング処理にかかる負荷が高く、高性能のマシンを要求する。このレンダリング処理にグリッドなどの分散コンピューティングを応用することで、より高精細な画像を生成しようとする試みも行われている。最近ではデジタルハリウッド大学院による「関ヶ原の合戦」映像制作プロジェクト（文献[16]）が行われている。

本研究では 3DCG の静止画像を対象としたレンダリングアプリケーションをグリッドによって構築する。画像フォーマットは 4K Digital Cinema を使用した。3D モデルはオープンソースのモデリングソフトウェアである Blender（文献[17]）を用いて作成し、オープンソースのレイトレーサーである YafRay（文献[18]）を用いてレンダリングを行う。

3. 概要

本研究で使用したグリッドシステムは、文献[3]で使用したシステムと同一のシステムである。このシステムを図 1 のように、仮想マシンをスレーブノードとなるマシンに導入した。なお、グリッドミドルウェアは Globus Toolkit 4 である。

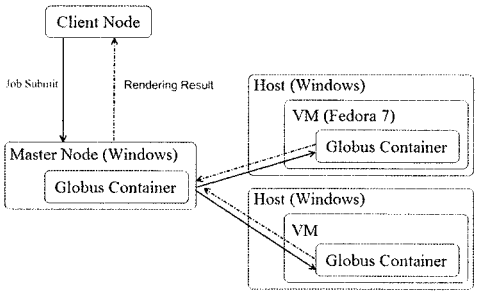

図 1 システム概念図

また、本システムにおけるジョブとは、Blenderによって作成された3DCGモデルのレンダリングである。

本研究においては、スレーブノードに使用されるマシンの性能は同一であり、ホストOSも同一のものを使用するという前提で評価を行った。また、仮想マシンに割り当てるIPアドレスはブリッジ接続による一意のアドレスでなければならない。本システムでは各スレーブノードに対して直接接続することが必要であり、NATなどを使用した場合にスレーブノードへの接続が不可能となるためである。

またマスターノードに関しては仮想マシンを用いず、直接Windows上で動作させている。マスターノードで行われる処理はスレーブノードと比較して負荷が低い処理であるからである。

4. 評価

4.1 評価環境

表1は本研究においてスレーブノードとして使用したマシンの性能である。このマシン4台を用いて評価を行った。また、仮想マシンはVMware PlayerおよびServerを用い、ゲストOSとしてFedora 7を使用した。

表1 マシンスペック

OS	Microsoft Windows XP Home Edition Version 2002 Service Pack 2
CPU	AMD Athlon X2 Dual Core Processor 4600+, 2.41 GHz
Memory	1.93 GB

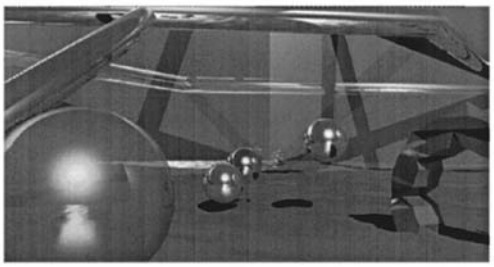

図2 評価画像

評価は図2に示された3DCG画像のレンダリングによって行った。図2の画像は4K Digital Cinemaフォーマットを用いた静止画像である。この画像を格子状に分割し、分割数ごとに応答時間を比較した。分割は2からはじめ、最大64分割とした。

評価に際しては比較のために、Windows Gridが使用するCPUコア数を1つに制限した場合と、2つとも使用した場合の評価も行った。ただし、CPUコア数2つの場合は、CPUコア数1つの場合でスレーブマシン4台とほぼ同等の性能を発揮することが文献[3]より明らかであるので、スレーブノードして使用したマシン台数は2台である。つまり、シングルコアの場合におけるマシン4台で構成されるグリッドは、デュアルコアマシン2台の場合と同等である。

また、仮想マシンからはCPUは1つにしか見えないという条件が付く。従って、各グリッドシステムが認識するCPUコア数は表2のようになる。特に仮想マシンを使用した場合では、グリッド環境が使用するCPUコア数は2つであるが、レンダラーであるYafRayからは1つのCPUコアのみが認識されていることである。なお、表中の表記のRMは実マシン上で動作させていることを示し、VMは仮想マシン上で動作させていることを示している。

表2 グリッドとCPUコア数の関係

	Single	Dual	VM
OS	Windows	Windows	Fedora 7
RM / VM	RM	RM	VM
CPU (Grid)	1	2	2
CPU (YafRay)	1	2	1

4.2 評価結果

図3に本実験の評価結果をプロットしたものを示す。なお、VMは仮想マシンによるLinuxグリッドを示し、Single, Dualはそれぞれ Windows Gridで使用したCPUコア数の設定の違いである。

評価結果による応答時間の変化傾向は文献[2]、[3]と同様の傾向がみられる。

通常の仮想マシン環境では少なからずオーバーヘッドが存在し、ホストOSよりも処理性能は低下する。しかし本評価の結果では、仮想マシンによるLinuxグリッドの方が同数のスレーブノードを使用した場合のWindows Gridよりも処理性能が高いことが示されている。

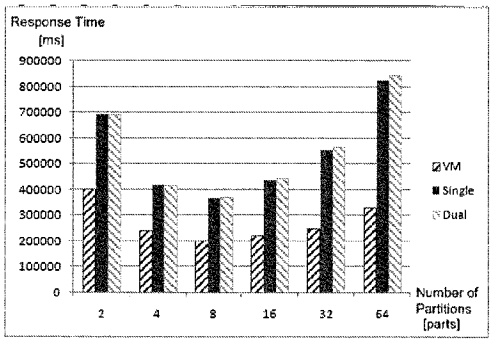

図3 評価結果

5. 考察

図3に示された評価結果は、同数のスレーブノードを使用する場合には Windows による実マシン上でのグリッドよりも、仮想マシンを用いた Linux グリッドの方が高い性能を発揮することを示している。通常の仮想マシン環境では、ゲスト OS の処理性能がホスト OS よりも高くなることはまず考えられない現象である。しかし、今回の実験環境はデュアルコアプロセッサによるグリッド環境である。そのため、シングルコアプロセッサで想定される場合とは違う現象が発生したと考えられる。

5.1 応答時間についての考察

文献[3]において、デュアルコアプロセッサを用いたグリッドは、同性能のシングルコアプロセッサ、もしくは一つの CPU コアを使用したグリッドよりも性能が向上することが実験的にわかっている。具体的には、デュアルコアマシン上に配置されたノードは、シングルコアマシン上に配置されたノードのおよそ倍の性能を発揮する。これと同様の現象が仮想マシン上のグリッドにおいても発生したものと考えられる。

図4は本研究で使用しているグリッドにおいて、画像サイズを 512x270 に縮小し、スレーブノードを1つとした場合の応答時間である。なお、図4における仮想マシン上のゲストOSは、Fedora 7 に問題が発生したため、Ubuntu 7.04 を使用した。

スレーブノードが1台の場合は、各グリッドシステムによる差異がより顕著に現れる。特に、デュアルコアを使用するよう設定した Windows グリッドと、仮想マシン上の Linux グリッドの応答時間が非常に近い。これは、仮想マシンにおけるグリッドがデュアルコアを使用した Windows ネイティブのグリッドと同等の性能を発揮したということである。

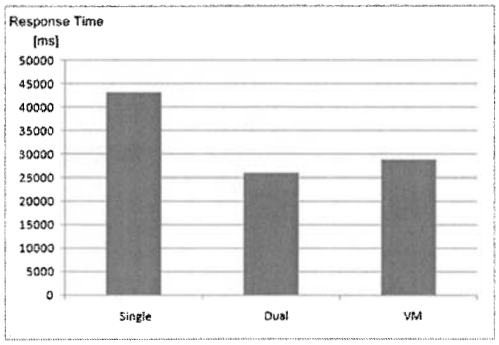

図4 スレーブノード1つの場合の応答時間

5.2 単位ジョブあたりの処理時間

図5は図3の評価時における、単位ジョブあたりの応答時間が画像分割に応じてどのように変化していったかを示したものである。

図5においては単位ジョブあたりの処理時間が、Windows グリッドのおよそ 1/2 となっている。これは Windows グリッドにおいて、レンダリング時に割り当てる CPU コアを2つにした場合と同様の変化である。したがって、仮想マシンを用いた場合 CPU に搭載されているコアを2つとも使用しているということを示している。

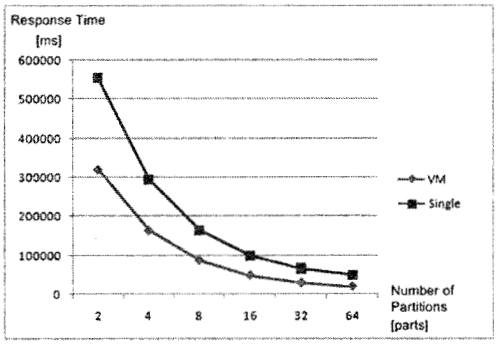

図5 単位ジョブあたりの処理時間変化

また図5からは、ジョブを小さくしていくに従い、処理時間の変化が少なくなっている様子が見られる。すなわち、ジョブが小さすぎる場合にはオーバーヘッドが無視できない状態となり、システム全体の性能が低下する。図3においてもジョブ分割数 64 の場合では、ジョブ 2 分割の場合よりも応答時間が長いことが確認できる。

5.3 仮想マシン上の Windows グリッド

仮想マシンを用いた Linux グリッドは、Windows PC のリソースを活用する場合に有効な手段であることがわかった。そこで、仮想マシン上で動作するゲスト OS を変更した場合にグリッドの性能が変化するかどうかを確認するために、ゲスト OS として Windows を用いた場合の評価を行った。図 6 はその結果である。なお、図中のラベル VM の括弧内はゲスト OS の名称である。また、"Windows Single"とは、シングルコアとして動作させた Windows グリッドである。

図 6 から仮想マシン上に構築された Windows グリッドは性能が低いことがわかる。

使用した仮想マシンソフトウェアは同一であるにもかかわらず、このような現象が発生した原因として考えられるのは、仮想マシンソフトウェアの特性である。

仮想マシンとして使用した VMware Player では、カーネルモード命令のみをエミュレートし、ユーザーモード命令は直接プロセッサによって実行されることは前述のとおりである。このことから、Windows OS 自体がカーネルモー

ド命令を使用する頻度が非常に高いものと推測される。そのためにオーバーヘッドが増加し、システム全体の性能が低下したと考えられる。

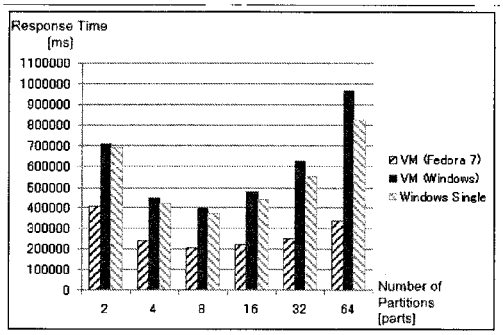

図 6 VM のゲスト OS による比較

6. おわりに

今回の評価では、仮想マシンがゲスト OS に提供する CPU は 1 つとした。その場合でも仮想マシン自体はデュアルコア上で動作している。したがって、仮想マシンがデュアルコアプロセッサを有効に活用することでゲスト OS の処理性能を向上していたと結論付けることが可能である。実際、1 つの CPU コアのみを使用する Windows グリッドでは、片方の CPU コアを占有していたが、仮想マシンによるグリッドを実行している場合は各 CPU コアに処理を分散させている状況がみられた。

本研究は Windows によるグリッドと、仮想マシンを用いた Windows 上における Linux グリッドとの比較を行ったものである。その結果、デュアルコアプロセッサを搭載した PC 上においては、単純に Windows グリッドを構築するよりも、仮想マシンを導入し Linux によって構築されたグリッドの方が高い処理性能を持つということが判明した。仮想マシンを用いることによって少なからずオーバーヘッドが生じるものの、デュアルコアによる並列処理による性能向上の効果が大きいということである。

また、Linux ではカーネルモード命令の発生頻度が Windows よりも低いと推測されることから、シングルコアプロセッサにおいても、エミュレーションによるオーバーヘッドの影響は少ないと考えられる。したがって、仮想マシンを用いてグリッドを構築することで、安価な計算資源を確保することが可能である。

参考文献

[1] 国立情報学研究所グリッド研究開発推進拠点、http://www.naregi.org/

[2] 田中堅一、“オープンソースグリッドによる CG の並列計算”、FIT2007 第 6 回情報科学技術フォーラム

[3] 田中堅一、“オープンソース Windows グリッドによる CG の並列計算”、第 15 回マルチメディア通信と分散処理ワークショップ

[4] BOINC, http://boinc.berkeley.edu/

[5] The Globus Alliance, http://www.globus.org/

[6] SETI@home, http://setiathome.berkeley.edu/

[7] Open Grid Forum, http://www.gridforum.org/

[8] EGEE (Enabling Grids for E-sciencE), http://www.eu-egee.org/

[9] The Globus Consortium, http://www.globusconsortium.org/

[10] Univa Grid Computing, http://www.univagridcomputing.com/

[11] Virtual Machine, http://en.wikipedia.org/wiki/Virtual_machine

[12] ITmedia エンタープライズ : 仮想マシンとは何か？、http://www.itmedia.co.jp/enterprise/articles/0612/15/news007.html

[13] VMware – Virtualization Software, http://www.vmware.com/ja/

[14] Microsoft Virtual PC 2007, http://www.microsoft.com/japan/windows/virtualpc/default.mspx

[15] Computer Laboratory - Xen virtual machine monitor, http://www.cl.cam.ac.uk/research/srg/netos/xen/

[16] 「関ヶ原の合戦」映像制作プロジェクト、http://digieco.dhmsl.jp/

[17] blender.org – Home, http://www.blender.org/

[18] Free Raytracing for the masses - YAFRAY.ORG, http://www.yafray.org/