社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－HPC－112 (9)
2007／9／9

ステートレス仮想クラスタの構想

小川宏高[†] 中田秀基[†] 横井威[†]
江原忠士[†] 谷村勇輔[†] 関口智嗣[†]

計算機資源の効率的な運用の方法として仮想化が注目されており，仮想的なクラスタを管理するさまざまなシステムが提案されている．我々も 2005 年度よりユーザからの依頼に応じて予約ベースで仮想クラスタを構築・提供するサービスを実現する AIST-SOA 仮想クラスタ管理システムを研究・開発している．しかし，多くのシステムは，仮想ストレージの managability や performance に問題を抱えている．例えば，仮想計算機のディスクイメージがホスト計算機に束縛されているために，耐故障性の実現や負荷分散が困難であったり，並列読み書き性能が不十分であったりする．こうした問題を解決するため，本稿ではホスト計算機・仮想計算機の全てのストレージを仮想ストレージサーバに集約し，ディスクレスシステムとして運用するステートレス仮想クラスタを提案し，そのコアコンポーネントの実装について説明する．

Conception of Stateless Virtual Clusters

Hirotaka Ogawa,[†] Hidemoto Nakada,[†] Takeshi Yokoi,[†]
Tadashi Ebara,[†] Yusuke Tanimura[†] and Satoshi Sekiguchi[†]

Virtualization techniques are gaining acceptance as a core technology for utilizing computing resources in computer centers, and several systems are proposed for contructing ang managing "virtual clusters". Since 2005, we also have been developping "AIST-SOA Virtual Cluster Management System" which allows users to reserve a virtual cluster through an easy-to-use Web interface. For each reservation, the system automatically allocates and configures a bunch of virtualized computing resources including VMware-based virtual machines, iSCSI storages, and networks for a virtual cluster. However, most systems have serious problems in managability and performance in *virtual storage*. For example, storages for virtual machines strictly binds to a hard disk owned by their host machine, therefore, it could be difficult to realize fault-tolerance and/or load-balancing and to accomplish enough parallel read-write performance. To resolve these problems, in this paper, we propose *stateless virtual cluster*, which concentrates all of storages for virtual and physical machines to virtual storage nodes, and enables diskless boot from them. And we also describe the design and implementation of core component of stateless virtual cluster.

1. はじめに

計算機センタやデータセンタなどに代表される計算機資源を集約的に保有する組織では，資源の稼働率の向上や設備コスト・運用コストの削減が強く求められており，近年こうした要請を満たす方法として仮想化技術が注目されている．仮想化技術とは，計算機システムの構成要素である CPU，ストレージ，ネットワークなどを論理的に複数に分割し，複数の CPU，複数のストレージ，複数のネットワークとして利用可能にする技術である．この技術を用いることで，物理的に保有している計算機資源をより多くの仮想的な計算機資源として利用することができ，運用の自由度や資源の稼働率の向上，設備コストの圧縮に大きな効果がある．

こうした仮想化技術の恩恵を広くまた簡便に享受するために，仮想的な計算機資源の集合（仮想クラスタ）を「ユーティリティ的」に，つまりユーザの要求に応じてオンデマンドかつフレキシブルに，構築する技術が強く求められている．

我々は 2005 年度よりユーザからの依頼に応じて予約ベースで仮想クラスタを構築・提供するサービスを実現する AIST-SOA 仮想クラスタ管理システム[1] を研究開発している．具体的には，このシステムを用いると，(1) ユーザはまず Web インタフェースを用いて利用する計算機資源の量（CPU の個数，メモリー容量，ディスク容量など）と構成情報，その占有時間を指定して予約し，(2) 予約開始時間が近づくと，システムは予約内容に基づいて，VMware Server 1.0 によ

† 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)

る仮想計算機，VLAN，iSCSI ストレージなど必要な計算機資源を生成・確保し，(3) その仮想クラスタに対して San Diego Supercomputing Center を中心に開発されているクラスタ自動構築ツールである NPACI Rocks Toolkit[2] を用いて OS・アプリケーションなどのインストールと設定を行ってユーザに提供する，ことができる．また，我々と目的を同じくしたエフォートとして，ORE Grid[3]，Virtual Workspace[4]，Xen-OSACAR[5]，Cisco 社の VFrame[6] などがある．

こうした仮想クラスタ構築技術でとりわけ重要な issue になると考えられるのは，仮想化されたストレージの managability と performance である．

仮想計算機のファイルシステムは，通常ホスト計算機のファイルシステム上のイメージファイル，もしくはパーティションを用いて実現されるが，この方法ではファイルシステムがホスト計算機に保有される物理ストレージ装置に強く束縛されてしまう．このため，ファイルシステムの容量設定の自由度が低くなり，動的な負荷分散や障害対策を目的としてファイルシステムを含んだ仮想計算機イメージをホスト計算機間でマイグレーションすることも難しくなる．また，管理対象となる物理ストレージ装置がホスト計算機群に分散して配置されるということは，管理自体を困難にする上，耐故障性を実現したければホスト計算機の責任で実現しなくてはならない．

この問題を解決する自明な方法の一つは，仮想計算機のファイルシステムを専用のストレージノードに集約することである．しかし．別の問題がある．一つは，ストレージノード上に置かれたファイルシステムイメージを，仮想計算機からいかにトランスペアレント利用可能にするかという問題であり，もう一つは，ストレージノードに要求されるストレージ容量やスループットをいかに達成するかという問題である．

本稿では，こうした問題を解決することを目的としたステートレス仮想クラスタの構想とそれに向けた「AIST-SOA 仮想クラスタ管理システム」の取り組み，今後の計画を述べる．

2. AIST-SOA 仮想クラスタ管理システム

ここでは AIST-SOA 仮想クラスタ管理システムの概要を説明する．

本システムには，クラスタプロバイダ，サービスプロバイダ，ユーザの三者が関与する．クラスタプロバイダは，本システムを使用して，所有する実クラスタを管理する主体である．クラスタプロバイダは，エンドユーザに対して直接サービスを提供することはない．クラスタプロバイダに対する顧客はサービスプロバイダである．サービスプロバイダは，エンドユーザに対してサービスを提供する主体である．サービスプロバイダは，計算資源を持たず，クラスタプロバイダと契約して仮想クラスタの提供を受け，その上に展開したサービスをユーザに提供する．

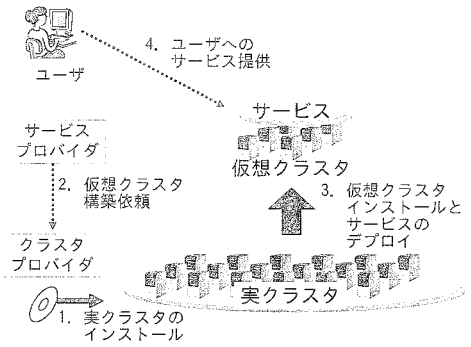

図 1 クラスタプロバイダ，サービスプロバイダとユーザ

仮想クラスタはフロントエンドノードと 1 つ以上の計算ノードから構成される．フロントエンドノードと計算ノードはプライベートアドレスのローカルネットワークで接続されている．クラスタプロバイダは，フロントエンドノードのグローバルネットワークへのインターフェイスの IP アドレスをサービスプロバイダに提供する．

以下に，本システムを利用したクラスタ提供のシナリオを示す (**図 1**)．

(1) まず，クラスタプロバイダは本システムを利用し，実クラスタをインストールする．

(2) 次にサービスプロバイダがクラスタプロバイダに対して仮想クラスタ構築を依頼する．その際にサービスプロバイダは，使用開始/終了時刻，使用計算機台数，必要メモリ量，ストレージ量，提供するサービスを構成するためのアプリケーションプログラムなどの情報を提供する．アプリケーションプログラムは，サービスプロバイダが用意する．

(3) クラスタプロバイダは本システムを用いて仮想クラスタを実クラスタ上に構築し，サービスプロバイダが用意したアプリケーションプログラムをインストールして，サービスをデプロイし，サービスプロバイダに提供する．

(4) サービスプロバイダはアプリケーションプログラムを利用したサービスを，ユーザに対して提供する．

実装の詳細については，1) を参照のこと．

3. 仮想クラスタ用ストレージの問題点

3.1 一般的な問題点

仮想クラスタ構築技術でとりわけ重要な issue になると考えられるのは，仮想化されたストレージの

managability と performance である．

今日 CPU やネットワークの仮想化技術自体はほぼ成熟したと言ってよいのに対して，ストレージに関してはそうではない．仮想計算機のファイルシステムは，通常ホスト計算機のファイルシステム上のイメージファイル，もしくはパーティションを用いて実現されるが，この方法ではファイルシステムがホスト計算機に保有される物理ディスク装置に強く束縛される．

このことは以下に示すさまざまな disadvantage をもたらす:

- ファイルシステムの総量がホスト計算機の物理ディスク装置に制約されるため，容量設定の自由度が低い．
- 動的な負荷分散や障害対策を目的としてファイルシステムを含んだ仮想計算機イメージをホスト計算機間でマイグレーションすることが困難になる．
- 物理ディスク装置がホスト計算機群に分散して配置されるため，管理対象が分散する．また，耐故障性を実現したければホスト計算機の責任で実現しなくてはならない．

上記の問題を解決する自明な方法の一つは，仮想計算機のファイルシステムを専用のストレージノードに集約することである．しかし．別の問題がある．

一つには，ストレージノードに低価格で汎用に利用が可能な NFS または iSCSI[7] を仮定すると，ストレージノード上に置かれたファイルシステムイメージを仮想計算機からいかにトランスペアレントに利用可能にするかという問題がある (managability)．つまり，仮想計算機がファイルシステムイメージを自動的にルートファイルシステムとして認識し，利用できるようなソリューションが必要である．

もう一つは，コモディティとして入手可能な CPU のコア数の増加などによって，単体のホスト計算機でサービス可能な仮想計算機の個数は増加しており，ストレージノードに要求されるストレージ容量やスループットをいかに達成するかという問題がある (performance)．言うまでもなく N 台規模の物理クラスタに M 個ずつの仮想計算機を立ち上げれば，NM 倍のストレージが必要になり，スループットの期待値は $1/NM$ 倍にまで低下するからである．

3.2 AIST-SOA 仮想クラスタ管理システムでの問題点

2 で述べた AIST-SOA 仮想クラスタ管理システムでは，仮想計算機の root ファイルシステムを iSCSI ターゲット (サーバ) 上の仮想化されたディスクボリュームを利用して実現する．

具体的には，まずホスト計算機が iSCSI イニシエータ (クライアント) となって，iSCSI ターゲットに接続する．すると，そのストレージはホスト計算機上ではブロックデバイスとして認識され，デバイス名 (/dev/sdc など) が与えられる．このデバイスを物理ディスクとして指定し，VMware Server を起動する．つまり，VMware Server としては通常の物理的なデバイスをマウントしている場合と同じ動作をしているにもかかわらず，ホスト計算機が仲介することで，実際には iSCSI 経由でターゲット上のストレージを使用できる．

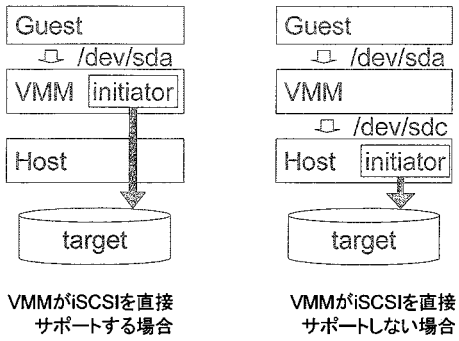

図 2 iSCSI の構成

図 2 にこの様子を示す．左図が，仮想計算機システムが iSCSI を直接サポートしている場合を示している．ゲスト OS のディスクアクセスは仮想計算機システムによって iSCSI プロトコルに変換される．これに対し，右図が今回とった構造である．ゲスト OS のディスクアクセスは，仮想計算機によってホスト計算機にアタッチされたディスクへのアクセスに変換される．これが，ホスト計算機にインストールされた iSCSI イニシエータによって，さらに iSCSI プロトコルに変換される．

この方法は 3.1 で述べた問題の多くを解決する．仮想計算機のファイルシステムがホスト計算機と分離され，iSCSI ストレージ上に格納されるため，容量設定の自由度が確保でき，負荷分散や障害対策を目的としてファイルシステムを含んだ仮想計算機イメージをホスト計算機間でマイグレーションすることも容易になる．また，管理対象がストレージノードに集中するため，ホスト計算機の責任で仮想計算機のディスクイメージの耐故障性を保証する必要がなくなる．

また，iSCSI は仮想計算機から見ると，ストレージの IP レベルインタフェースを規定しているだけなので，性能や耐故障性の実現を下位レイヤーに隠蔽することができる．例えば，複数の RAID コントローラを装着したストレージノードを複数用意して，冗長化と分散化を行ったストレージクラスタとして利用することで耐故障性と並列読み書き性能を達成することも可能である．

しかし，ホスト計算機の物理ディスクの使用を前提としているため，ホスト計算機の設定内容がそのホスト計算機の物理ディスク装置に束縛される．したがって，そのディスク装置に障害が起きた場合，仮想計算

機イメージは失われないがホスト計算機の設定内容は失われる可能性がある．また，ホスト計算機を他の業務のために明け渡す必要に迫られた場合にもマイグレートすることは困難である．

また，ホスト計算機が仮想クラスタのためにドミナントに利用可能でない場合（例えばホスト計算機が通常時は業務に使用されていて仮想クラスタにはストレージを資源として貸し与えることができない場合）や，省電力のためディスクレス構成のホスト計算機を利用したい場合，そもそも「ホスト計算機の物理ディスク」の存在が仮定できず，1) での我々のアプローチは適用できない．

4. ステートレス仮想クラスタ

3 で述べた問題を解決すべく，我々は「ステートレス仮想クラスタ」を提案する．

ステートレス仮想クラスタの基本的なアイディアは，ホスト計算機と仮想計算機の双方をディスクレスブートすることで，仮想クラスタシステム全体の，物理ディスク装置への依存を一掃することである．システムへの要件は，ホスト計算機と仮想計算機の双方が PXE compliant であるということだけである．現代のほとんどの PC・サーバ製品に付属している NIC も PXE compliant であり，また，VMware Server の仮想 NIC もそうである．AIST-SOA 仮想クラスタ管理システムが利用している Rocks もまた PXE compliant な NIC を要求することからこの要件はそれほど強い制約ではない☆．

ステートレス仮想クラスタでは，**図 3** に示すように，まずホスト計算機群をディスクレスブートするようにセットアップし，次にそのホスト計算機上で動作する仮想化ソフトウェアを使って仮想計算機群を生成する．さらにその仮想計算機群もディスクレスブートするようにセットアップする．

このような構成を採ることで，ホスト計算機と仮想計算機のファイルシステムがホスト計算機の物理ディスクから分離され，iSCSI ストレージ上に格納されるため，容量設定の自由度が確保できる．また，負荷分散のためにファイルシステムを含んだ仮想計算機イメージをホスト計算機間でマイグレートしたり，ホスト計算機の障害時には代替のノードにまるごとマイグレートしたりすることができる．

3.2 の前半で触れたように，1) の実装では，仮想計算機が利用する仮想ディスクをアタッチする目的でホスト計算機が iSCSI イニシエータとなっていた．これに対して，ステートレス仮想クラスタではホスト計算機・仮想計算機がそれぞれ自分の利用するストレージをアタッチする目的で iSCSI イニシエータとなる．したがって，両者の独立性が高まり，管理の容易性も増す．

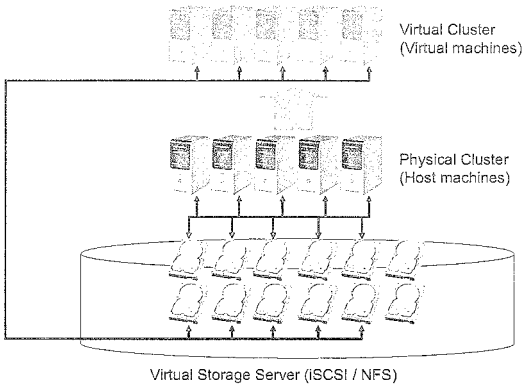

図 3 ステートレス仮想クラスタの概要

また，ホスト計算機の物理ディスクは，原理的にはホスト計算機および仮想計算機のスワップファイル，ディスクキャッシュ，あるいは一時的なワークスペースとして利用することも可能なため，必ずしも無駄とはならない．

以下では，ディスクレスブートの実現方法について説明した後，ステートレス仮想クラスタを実現するコアコンポーネントとなる，開発中である iSCSI/NFS Diskless Roll の実装について述べる．

4.1 ディスクレスブート

ディスクレスブートは，PXE ブートしたカーネルから NFS root や iSCSI デバイスを root mount することで，ディスクレスの状態でマシンを動作させるものであり，特に NFS root を用いる方法は比較的枯れた技術として知られている．

図 4 に示すように，Diskless Node は，Linux がサポートしている 2 段階のブートプロセスを利用して実現される．

まず，Frontend で動作する DHCP サーバと TFTP サーバから，それぞれ IP アドレスとネットマスク，kernel イメージと RAM ディスクイメージ (initrd) を取得する．次に kernel イメージをメモリ上に読み込み，initrd を初期 root ファイルシステムとしてマウントし，初期ブートプロセスを開始する．

初期ブートプロセスは initrd に書かれている通りに実行される．具体的には，必要なデバイスドライバなどを実現する kernel モジュールを読み込み，IP アドレスなどを設定し，NFS root を利用する場合には NFS のクライアントモジュール，iSCSI root を利用する場合には iSCSI イニシエータをそれぞれ起動する．続いて，ストレージノードが提供する NFS ディレクトリや，iSCSI ディスクを/tmproot にマウント

☆ ただし，Xen では DomainU を PXE ブートできない．この場合には，3.2 で述べたのと同様に Domain0 に一旦リモートストレージをアタッチする必要がある．

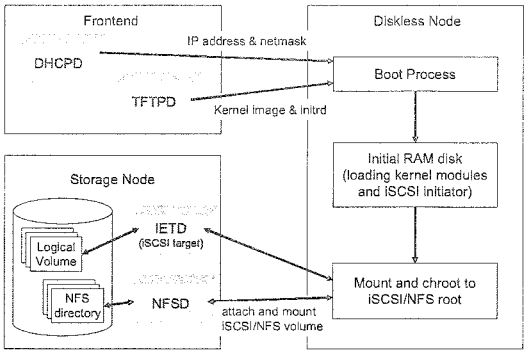

図 4 ディスクレスブート

し，そのディレクトリに chroot する．最後に chroot した root ファイルシステムから 2 段階目のブートプロセスを実行する．

4.2 実　　装

ステートレス仮想クラスタのインストールおよびブートシーケンスは，ホスト計算機と仮想計算機で共通化し，階層的に適用することができる．特に，1) の実装では両者とも Rocks を用いて管理・インストールされるため，Rocks の Roll として NFS と iSCSI によるディスクレスブート機能を実装することが望ましい．

実装において重要なのは，Rocks ではインストール時に PXE ブートを利用するのに加え，ディスクレスブート時も PXE ブートを行うということである．それゆえ，Frontend で動作する DHCP サーバと TFTP サーバは，インストール時とディスクレスブート時にそれぞれ適切な kernel イメージや initrd などを選択して PXE クライアントに渡すように設定されなければならない．

これに留意して行った実装の概要を**図 5** に示す☆．

まず，インストール時は以下のように実行される．

(1) Frontend ノードで insert-ethers を実行する．このとき，insert-ethers のプラグインモジュールがストレージを確保する．iSCSI の場合には iSCSI ターゲット上でディスク領域を確保し，ディスクレスノード用に公開し，NFS の場合には NFS サーバ上でディスクレスノード用のディレクトリを作成・公開する．

(2) 続いてディスクレスノードの電源を入れると，Frontend の DHCP サーバと TFTP サーバから，それぞれ IP アドレスとネットマスク，kernel イメージと initrd を取得し，初期ブートプロセスを開始する．

(3) ディスクレスノードは initrd に書かれたスクリプトに従い，Frontend 上の kickstart.cgi が生成するディスクレスノード用のカスタマイズされた kickstart ファイル (ks.cfg) をダウンロードし，anaconda インストーラでインストールを開始する．

(4) anaconda では，iSCSI ターゲットをアタッチ，または NFS マウントしてインストールプロセスを実行し，実行が完了したら，Frontend の XMLRPC サーバを呼び出す．

(5) XMLRPC サーバは続くディスクレスブートに必要な kernel オプションを DHCP サーバに，kernel イメージと initrd を TFTP サーバにそれぞれ設定・格納する．

ディスクレスブート時は，インストール時に設定された kernel オプションなどにしたがって，図 4 と同様の手続きにしたがってブートする．

4.3 Rocks による実装の拡張

Rocks でのインストールは，インストール対象のホスト計算機・仮想計算機にリモートストレージをアタッチした状態で行なわざるを得ない．しかし，実質的にはディスクレスブートのインストールはどのマシンにアタッチして実施しても構わない．インストール専用マシンにリモートストレージをアタッチし，適当なディレクトリにマウントした状態でコマンドラインで anaconda を実行してインストールを実施した後，そのリモートストレージをディスクレスノードに割り当てることもできる．

この方法を採った場合，インストール専用ノード，ディスクレスノードとも再起動する必要はないのでインストール時間の大幅な短縮が見込める．反面，インストール時にデバイスの probe，デバイスドライバの組み込みなどを正常に行なえない危険性はある．

仮想計算機のように均質であると考えてよいターゲットマシンに対しては，上記のようなインストール方法が極めて有効だと考えられる．

5. 関 連 研 究

Nopparat Nopkuat らの Diskless Roll[8] は NPACI Rocks で，Thin-OSCAR[9] は OSCAR クラスタインストーラで，それぞれディスクレスブートを実現するものである．両者の実装は，NFS root での利用に限られており，仮想クラスタへの利用は考えられていない．また，後者はディスクレスブート用の initrd を手動で生成する必要があるなど，実装が不十分な点もある．

6. ま と め

計算機資源の効率的な運用の方法として仮想化が注目されており，仮想的なクラスタを管理するさまざまなシステムが提案されている．しかし，多くのシステム

☆ 図は iSCSI の場合．NFS を用いる場合にもほぼ構造は同じであるため，割愛した．

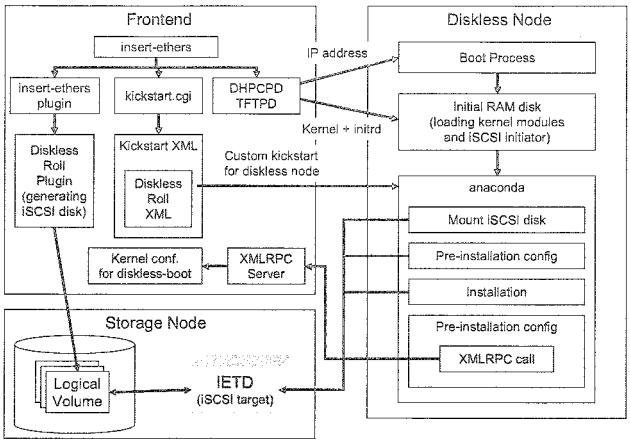

図 5 iSCSI Diskless Roll

は，仮想ストレージの managability や performance に問題を抱えている．例えば，仮想計算機のディスクイメージがホスト計算機に束縛されているために，耐故障性の実現や負荷分散が困難であったり，並列読み書き性能が不十分であったりする．

こうした問題を解決するため，ホスト計算機・仮想計算機の全てのストレージを仮想ストレージサーバに集約し，ディスクレスシステムとして運用する「ステートレス仮想クラスタ」を提案した．また，AIST-SOA 仮想クラスタ管理システムを用いて，ステートレスな仮想クラスタシステムを実現するためのコアコンポーネントの実装について説明した．

スペースの都合で割愛したが，1000Base-T のスイッチに直結された 2 ノードを用いての性能評価も行った．仮想計算機に直接アタッチされた iSCSI ディスクを利用する場合と，1) のようにホスト計算機のイニシエータにアタッチした iSCSI ディスクを間接的に利用する場合では，インストール時間に大きな差はなかった．

今後の課題はたくさんある．まず，実装の完成度を上げ，実用的な規模での性能評価を行うことである．また，ステートレス仮想クラスタはストレージサーバの並列読み書き性能が十分に達成できることを前提としているが，それが比較的安価かつ容易に実現できるようなシステム設計も必要である．

謝　辞

Diskless Roll のオリジナル開発者である Thai National Grid Center の Nopparat Nopkuat に感謝する．

参 考 文 献

1) 中田秀基，　横井威，江原忠士，谷村勇輔，小川宏高，関口智嗣：仮想クラスタ管理システムの設計と実装，情報処理学会論文誌 ACS (to appear) (2007).
2) Papadopoulos, P.M., Katz, M.J. and Bruno, G.: NPACI Rocks: Tools and Techniques for Easily Deploying Manageable Linux Clusters, *Cluster 2001: IEEE International Conference on Cluster Computing* (2001).
3) 高宮安仁，山形育平，青木孝文，中田秀基，松岡　聡：ORE Grid: 仮想計算機を用いたグリッド実行環境の高速な配置ツール，先進的計算基盤システムシンポジウム SACSIS2006 論文集，pp. 351–358 (2006).
4) Keahey, K., Foster, I., Freeman, T. and Zhang, X.: Virtual Workspaces: Achieving Quality of Service and Quality of Life in the Grid, *Scientific Programming Journal* (2006).
5) Vallée, G. and Scott, S.: Xen-OSCAR for Cluster Virtualization, *Workshop on XEN in HPC Cluster and Grid Computing Environments (XHPC '06)* (2006).
6) : Cisco VFrame Server Fabric Virtualization Software, `http://cisco.com/en/US/products/ps6429/index.html`.
7) : iSCSI Specification. `http://www.ietf.org/rfc/rfc3720.txt`.
8) Nopkuat, N. et al.: Diskless Roll, `http://research.thaigrid.or.th/en/Diskless_Roll`.
9) Ligneris, B. and Giraldeau, F.: Thin-OSCAR : Design and future implementation, *Proc. of 17th Intl.Symp. on High Performance Computing Systems and Applications*, pp. 261–265 (2003).