仮想クラスタ管理システムの設計と実装

中田秀基[†] 横井威[†] 江原忠士[†,††]
谷村勇輔[†] 小川宏高[†] 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システムが提案されている．しかしこれらのシステムは，クラスタを単なるノードの集合と考えており，クラスタとしての統合的な運用に必要となるさまざまな機能を実現していない．また，計算機資源を仮想化しているが，ストレージ，ネットワークを含めた仮想化環境を提供していない．我々は，これらの問題点を解決した仮想クラスタ管理システムを提案する．本システムは，クラスタ構築システム Rocks を用いることで，クラスタ運用に必要なソフトウェアを整合して配置する．また，ストレージ資源を IP SAN 技術の 1 つである iSCSI を用いて仮想化，ネットワーク資源をタグ付き VLAN を用いて仮想化することで，管理コストが低く，安全な仮想クラスタ環境を実現する．

The Design and Implementation of a Virtual Cluster Management System

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

To fully utilize resources in computer center, virtualization techniques are getting popular and several systems are proposed for this purpose. However, they just provide set of virtualized nodes, not the 'virtual clusters'; i.e., they are not able to install and configure middlewares and tools that makes 'set of nodes' into 'cluster'. Another problem is that they just virtualize nodes, leaving storage resources and networks, which are equivalently essential for clusters, un-virtualized. we propose a virtual cluster management system which virtualizes compute resouces, as well as disk storage and network, and install and setup softwares that are essential for cluster operation, using Rocks, a cluster provisioning system. We vitualize storage with iSCSI and network with tagged VLAN.

1. はじめに

計算機センタやデータセンタなどの計算機資源を集中管理する組織において，資源の効率的運用を実現する方法として仮想化が注目されている．計算機資源やストレージ資源を仮想化して提供することによって，運用自由度と管理の容易性をともに向上させることができる．これによって資源を有効に活用し，ひいてはコストの低減につなげることが可能となる．

仮想クラスタを構築する際に注意すべき点の 1 つは，「仮想クラスタ」が単なる「仮想ノード」の集合ではないことである．クラスタがクラスタとして運用可能であるためには，名前空間やファイルシステム，ソフトウェアの共有が必要なだけでなく，モニタリングやバッチスケジューリングシステムなどの運用ソフトウェアが適切にインストール，コンフィギュレーションされていなければならない．

もう 1 つの重要な点として，ストレージとネットワークの仮想化があげられる．ストレージ資源は計算機資源と同様に重要なクラスタの要素であり，ストレージ資源も含めて仮想化しなければ，仮想クラスタのメリットである運用の自由度を得ることはできない．また，物理ネットワークを共有する複数の仮想クラスタを安全に分離するためには，ネットワークの仮想化も必要である．

我々は，上記の点に留意し，ソフトウェアが整合性をもって設定され，仮想化されたストレージとネットワークを持つ仮想クラスタを構築するシステムを提案する[1)]．本システムは，カスタマ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．VMware Server[2)] を用いて計算機を仮想化し，IP SAN（Stor-

† 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)
†† 数理技研
SURIGIKEN Co., Ltd.

age Area Network）技術の 1 つである iSCSI[3] を用いてストレージ資源を仮想化，タグ付き VLAN を用いてネットワークを仮想化する．このようにして構成された仮想計算機群をクラスタインストールシステム Rocks を用いてインストールすることで，運用ミドルウェアが整合してインストールされた，仮想クラスタを構成することができる．

次章以降の構成は以下のとおりである．2 章では，本システムで利用するクラスタインストールシステム Rocks に関して説明する．3 章で本システムに対する要件を整理し設計を議論する．4 章で実装を述べる．5 章で関連研究を，6 章で結論と今後の課題を述べる．

2. NPACI Rocks

本章では，本システムで使用するクラスタインストールシステム NPACI Rocks[4),5)] について述べる．

2.1 概　　要

Rocks は，NPACI（National Partnership for Advanced Computational Infrastructure）の一環として SDSC（San Diego Supercomputer Center）を中心に開発されたクラスタ管理ツールである．クラスタのノード群に対して一括で同じソフトウェアパッケージをインストールすることができる．OS としては，Red Hat Enterprise Linux をベースとした CentOS（Community ENTerprise Operating System）を使用する．

Rocks の対象となるクラスタの構造を**図 1** に示す．クラスタはフロントエンドと計算ノード群で構成される．フロントエンドと計算ノード群はプライベートなローカルネットワークを共有する．フロントエンドはこのローカルネットワーク向けと，グローバルネットワーク向けの 2 つのネットワークインタフェースを持ち，計算ノード群のルータとしても機能する．

Rocks は，個々のノードにソフトウェアをインストールするだけでなく，クラスタ運用のためのソフトウェアのインストール，設定も行う．代表的なものとして，NIS と類似した Rocks 独自のユーザ名空間管理サービスである 411[6]，クラスタのモニタリングを行う Ganglia[7] が設定される．

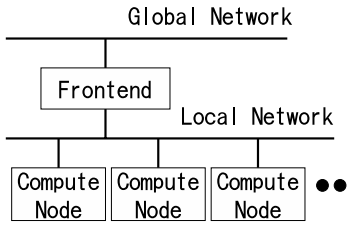


図 1 Rocks が管理するクラスタの構造
Fig. 1 Rocks cluster configuration.

2.2 Roll とアプライアンス

Rocks では Roll と呼ばれるメタパッケージによってアプリケーションを管理することができる．Roll には，RPM 形式のパッケージと，パッケージ間の依存関係，パッケージインストール後のデプロイ処理を記述することができる．依存関係と後処理は，XML で表現されるグラフ構造で指定する．クラスタ管理者は Roll を新たに追加することで，クラスタに新たな機能を追加することができる．現在，HPC 関連，グリッド関連のさまざまな Roll が Rocks チームやベンダから提供されている．

また，クラスタ内の個々のノードに対して，異なる構成でのインストールを行うことができる．個々のノード構成をアプライアンスタイプと呼ぶ．たとえば，クラスタ内に Web サーバ群とデータベースノード群をインストールする場合には，Web サーバアプライアンスとデータベースノードアプライアンスの 2 つを定義し，個々のノードをインストールする．

アプライアンスは Roll と直交する概念で，あるアプライアンスが複数の Roll によって規定されることもありうる．また，1 つの Roll が複数のアプライアンスの定義に寄与することもある．

2.3 Rocks によるクラスタのインストール

Rocks によるクラスタのインストールは下記のように行われる．まずクラスタ管理者は，フロントエンドを CD-ROM からインストールする．この際に，使用する Roll を指定する．次に，アプライアンスを指定して計算ノードを起動していく．この際に計算ノードの BIOS でネットワークブートを指定する．するとフロントエンドがインストールサーバとなり，計算ノードに自動的に OS とソフトウェアがインストールされる．この際，ノード名は計算ノードを起動した順番で自動的に割り当てられる．

3. 設　　計

3.1 仮想クラスタ利用シナリオ

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

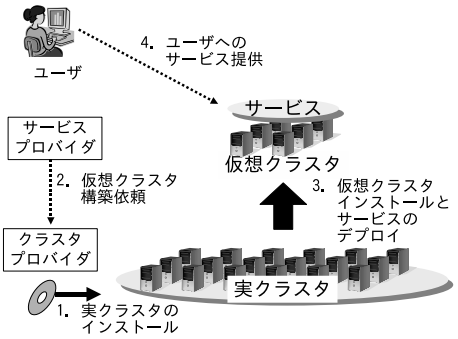


図 2 クラスタプロバイダ，サービスプロバイダとユーザ
Fig. 2 Cluster provider, service provider and user.

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．

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フェースの 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 2**）．

(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．

想定される使用法の 1 つとして，計算ファームがあげられる．短期的に膨大な量の計算をしなければならない会社があると仮定する．この会社はサービスプロバイダとして，1 月の間 100 台の計算機の使用する契約をクラスタプロバイダとかわす．予約した時刻には，仮想的なクラスタで Condor[8] などのジョブスケジューリングシステムが準備され，フロントエンドノードが引き渡される．これで，会社内の研究者がユーザとなり，計算をサブミットすることができるようになる．Condor のフロッキング機構を利用すれば，個々の研究者はなんら設定を変更することなく増大した計算パワーを享受することができる．

他の想定使用法としては，アプリケーションプログラムのテスト環境としての使用や，Web アプリケーション提供環境としての使用が考えられる．

また，教育用の計算機システムとして，科学技術計算用のクラスタと，授業用の環境を共存させることも考えられる．授業時間にあわせて毎週定時に仮想クラスタを構築し提供する．

3.2 仮想クラスタへの要請

本節では，仮想クラスタへの要請を整理する．

3.2.1 ノード構成

仮想クラスタは，それを利用するサービスプロバイダにとって，通常の物理的実体を持つクラスタと同じように見えていなければならない．クラスタの典型的な構成は，図 1 に示すように，1 つのフロントエンドノードに対して複数の計算ノードが接続された構成となる．多くの場合，フロントエンドノードのみが外部ネットワークにアクセス可能であり，計算ノードは内部のプライベートネットワークのみに接続される．計算ノードは，フロントエンドノードを経由して外部ネットワークへのアクセスを行う．

3.2.2 クラスタ運用ソフトウェアのインストール

クラスタを単なるノードの集合以上のものにするには，クラスタ運用のためのソフトウェアがインストールされていなければならない．このようなソフトウェアとしては，ユーザ名空間を管理する NIS や，クラスタをモニタする Ganglia[7]，ジョブスケジューリングシステムの Grid Engine，TORQUE，Condor などがあげられる．

これらのソフトウェアは，フロントエンドノードおよび計算ノードに，個別にインストールするだけでは意味がない．たとえば Grid Engine では，フロントエンドノードには計算ノードのリストが登録されていなければならず，計算ノードにはフロントエンドノードが登録されていなければならない．

3.2.3 計算機の仮想化

クラスタを仮想化するにはまず計算資源を仮想化しなければならない．この際，単一の実計算機上に複数の仮想計算機が起動でき，さらに，それらの間での

CPU 使用率の割当てが制御できることが望ましい．

3.2.4 ストレージの仮想化

クラスタが使用するストレージには，各ノードが利用するローカルストレージと，クラスタ全体として利用する共有ストレージがある．いずれの場合にも，サービスプロバイダの要請に応じて容量を自由に設定できることが望ましい．また，仮想クラスタ管理システム全体に対して，ストレージの管理が容易で，動的な追加が可能でなければならない．

通常，仮想計算機のファイルシステムは，ホスト計算機のファイルシステム上のファイル，もしくはパーティションを用いて実現する．しかし，この方法ではファイルシステムが，ホスト計算機に強く束縛されてしまい，容量設定の自由度が低い．また，仮想計算機のマイグレーションを行うことが難しい．さらに，ストレージ資源の管理という側面から考えても，すべてのホスト計算機にストレージが分散してしまい，管理が困難になることから好ましくない．ストレージは専用のストレージノードに集約できることが望ましい．

まとめると，ストレージの仮想化には，1) ストレージを位置透過に利用可能にし，2) 物理的なストレージ媒体にしばられない管理を実現する技術が必要になる．これらの，要件を満たすストレージ関連技術として，一連の SAN（Storage Area Network）技術がある．SAN は一般には仮想化技術というコンテクストでとらえられることは少ないが，上述の条件を満たすことからストレージの仮想化技術として使用することが可能である．

3.2.5 ネットワークの仮想化

複数の仮想クラスタが実クラスタ上で共存する場合，ネットワークのセキュリティを考慮する必要がある．標準的な仮想計算機構成では，仮想計算機はブリッジ接続を用いて，ホストとなる実計算機の所属するネットワークを共有する．この延長として仮想クラスタを構築すると仮想クラスタと実クラスタがネットワークを共有することになる．これは，実クラスタ上に複数の仮想クラスタを構成した場合，すべての仮想クラスタがネットワークを共有してしまうことを意味する．

しかし，多くのクラスタ管理者は，クラスタの属するローカルネットワークに対して，外部ネットワークよりも安全な環境を期待している．この期待にこたえるためには，仮想クラスタ間のネットワークを分離する必要がある．

3.3 設計の概要

3.3.1 ソフトウェアのインストールと設定

本システムでは，ソフトウェアのインストールと設定に NPACI Rocks を用いる．これは，Rocks がすでに幅広く利用されており，安定性に定評があることと，多くの Roll が各所から提供されているため，ユーザにとっての利便性が高いと思われるためである．

3.3.2 計算機の仮想化

計算機の仮想化には VMware Server [2] を用いた．VMware Server は VMware, Inc. が開発した仮想化ソフトウェアで，ライセンスの範囲で無償で使用できる．VMware Server は有償の VMware Workstation からいくつかの機能を省き，コンソール機能を切り離したものである．BIOS を含んだ完全な仮想化を提供するため，ゲストとして動かす仮想計算機の変更を要求しない点が特徴である．

VMware server では，仮想計算機間の CPU 割当てなどを指定することはできないが，プロセスの nice 値を制御することである程度の優先制御をすることができる．

3.3.3 ストレージの仮想化

3.2.4 項で議論したとおり，ストレージの仮想化技術として，Fiber channel や infini band などの高速ネットワーク技術を用いた SAN（Storage Area Network）を利用することができる．これらの技術を用いると，ストレージを集中管理することで運用コストの低減を実現することができるが，一般に非常に高価で，利用者が限定される．

我々は，低価格な実装が可能な IP SAN として普及が期待されている iSCSI [3] を用いる．iSCSI は，古くから外部デバイスのインタフェースに用いられてきた SCSI プロトコルを IP 経由で実現するものである．iSCSI を用いたクラスタでは，計算に使用するネットワークとストレージアクセスに使用するネットワークを共有することになるため性能低下が懸念されるが，文献 9) によれば問題ない範囲である．

iSCSI ではストレージを使用する計算機を initiator，提供する計算機を target と呼ぶ．一般に仮想計算機のファイルシステムを iSCSI を用いて構築するには，仮想計算機が initiator 機能をサポートしていなければならない．しかし，VMware Server はこの機能をサポートしていないため，単純な方法では，これを実現することができない☆．

我々は，これを次のような方法で解決した．まずホスト計算機が initiator となって，target に接続する．するとそのストレージはデバイス名（/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

☆ VMware Server ESX ではこの機能がサポートされている．

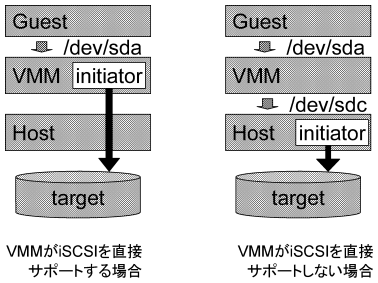


図 3 iSCSI の構成
Fig. 3 iSCSI configuration.

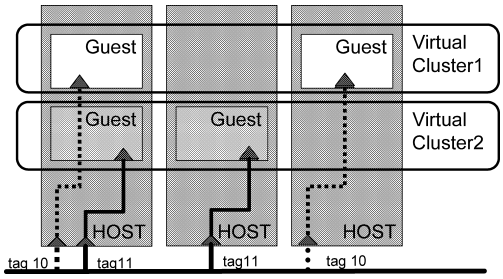


図 4 VLAN を用いたネットワークのセパレーション
Fig. 4 VLAN configuration.

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 target 上のストレージを使用できるのである．

図 3 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 initiator によって，さらに iSCSI プロトコルに変換される．

3.3.4 ネットワークの仮想化

ネットワークの仮想化手法としては，VLAN と VPN があげられる．前者は，さらにタグ VL とポート VLAN に分けられる．タグ VLAN はパケットにタグをつけ，スイッチでこれを認識することでルーティングを制限する手法，ポート VLAN はスイッチのポートごとにネットワークを固定的に割り当てる方法である．VPN は，主に広域ネットワークをまたがって LAN を構成することを目的とした技術で，パケットを暗号化カプセル化して，再度パケットを構成してルーティングを行う．VPN によるネットワークの仮想化は安全性が高いが，カプセル化によるオーバヘッドが非常に大きい．

我々はタグ VLAN を用いて，ネットワークの仮想化を実現する．個々の仮想クラスタに対してユニークな VLAN ID を割り当てる．ホスト計算機上ではゲストが属する仮想クラスタの VLAN ID を持つインタフェースを動的に作成し，そのインタフェースに対してブリッジ接続を行う．この様子を**図 4** に示す．3 つのホスト上に，それぞれ 2 つの仮想ノードを持つ仮想クラスタ 1，2 が構築されている．仮想クラスタ 1 に対してタグ 10 が，2 に対して 11 が割り当てられている．左端のホストは，2 つの仮想クラスタノードをホストしているため，2 つのタグつきネットワークをインタフェースを持つ．

この手法の特筆すべき点は，タグづけがホスト内部で生じるため，ゲスト上での設定がなんら必要ないことである．

4. 実　　装

4.1 システムの動作の概要

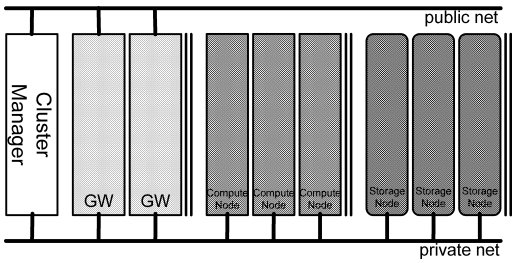
本システムのユーザであるサービスプロバイダは，本システムに対して，Web インタフェース，もしくは Web Service インタフェースを通じて仮想クラスタの予約を行う．この際に，サービスプロバイダは利用する計算ノードの数，メモリ量，各ノードのストレージ量，インストールするべきソフトウェア（Roll），ログインに使用する ssh の公開鍵などを指定する．

予約のポリシは first-come，first-served となっている．先行するリクエストによって資源が予約された結果，後続するリクエストの予約時間帯に予約条件を満たすだけの資源が確保できない場合，後続するリクエストは拒否される．

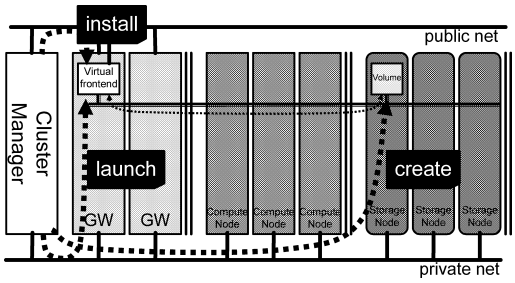
本システムは予約時間が到来した時点で，仮想クラスタを起動する．仮想クラスタは，Rocks でインストールするため，Rocks のインストールシーケンスを仮想クラスタ上で実現することになる．具体的には，仮想的なフロントエンドを構築し，その後仮想的な計算ノードを起動することで，仮想フロントエンドから仮想計算ノードへのインストールが行われる．

また，本システム全体も Rocks で管理されており，物理ノード群はすべて Rocks でインストールされる．したがって，本システムを利用する管理者は，物理クラスタの管理においても Rocks の恩恵を受けることができる．

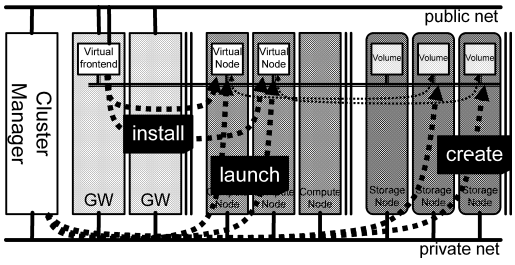
次に，システムの構成を詳しく見る．本システムは 4 種類の物理ノードから構成される（**図 5** A）．



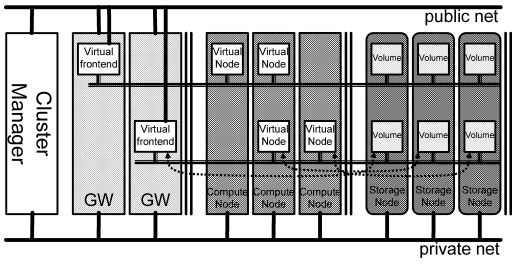
A. 初期状態



B. 仮想フロントエンドのインストール



C. 仮想計算ノードのインストール



D. 複数の仮想クラスタの実装

図 5 仮想クラスタのインストール
Fig. 5 Virtual cluster installation.

- **クラスタマネージャノード**
  実クラスタ，仮想クラスタ全体を統括するクラスタマネージャが稼動するノード．外部ネットワークと内部ネットワークの双方に接続を持つ．サービスプロバイダがアクセスする Web インタフェース，Web Service インタフェースはこのノード上に実装される．
  クラスタマネージャは物理クラスタをインストールする際の Rocks フロントエンドとして機能する．すなわち以降の 3 種類のノードは，クラスタマネージャをフロントエンドとして Rocks によって初期インストールが行われる．
- **ゲイトウェイノード**
  仮想フロントエンドを実行するためのノードクラスタマネージャノード同様に，外部ネットワークと内部ネットワークの双方に接続を持つ．
  機能としては仮想計算ノードの起動とネットワークの設定しか持たない．
- **物理計算ノード**
  仮想計算ノードを実行するためのノード内部ネットワークにのみ接続を持つ．機能としては仮想計算ノードの起動とネットワークの設定しか持たない．ゲイトウェイノードとの違いはネットワークインタフェースの数のみである．
- **ストレージノード**
  iSCSI によってストレージを提供するノード．大容量のディスクを持ち，仮想クラスタに提供する機能を持つ．内部ネットワークに対してのみ接続を持つ．
  ストレージ領域の管理には，LVM（Logical Volume Manager）を用いる．ストレージ割当てのリクエストに対して，LV を新たに生成し iSCSI のボリュームとして公開する．LVM を用いることで，物理ディスクのサイズや境界にとらわれずに任意のサイズのストレージを割り当てることが可能になる．

Rocks は，バージョン 4.2.1 をベースに，Rocks チームが改変し，4.3.2 で述べる自動フロントエンドインストール機能を追加したものを用いた．VMware Server にはバージョン 1.0.1 を用いた．

4.2 クラスタマネージャの実装

クラスタマネージャはクラスタマネージャノード上で稼動するデータベースにアクセスする一連の python スクリプトで構成されている．本システムの持つ状態は，すべてこのデータベースで管理されており，ファイルやメモリ上で管理している情報はない．データベース上には大別して，2 つの情報が納められている．資源タイムテーブルと仮想クラスタ予約テーブルである．

資源タイムテーブルには，物理クラスタ上のすべての資源の時間軸に沿った使用予定が納められる．具体的には，ゲイトウェイノードおよび物理計算ノード上のメモリ量，ストレージノード上の物理ボリュームの量，VLAN のタグを資源と考え，それぞれに対するタイムテーブルを保持している．

仮想クラスタ予約テーブルには，仮想クラスタの

リクエストの情報が納められる．予約の入力動作はhttpd サーバの CGI として機能するスクリプトとして記述されており，このスクリプトが仮想クラスタ予約テーブルを更新する．予約リクエストを処理するスクリプトは，データベースを検索して，リクエストされた時間帯にリクエストを満たすだけのメモリ，ストレージなどの資源があるかどうかを調べる．リクエスト満たすことができる場合には資源タイムテーブルをアップデートして，資源を確保すると同時に，インストールするべきアプリケーションや仮想クラスタに設定する ssh の公開鍵などの，仮想クラスタの構築に必要な情報をデータベースの仮想クラスタ予約テーブルに書き込む．

仮想クラスタのインストールを実行するするスクリプトは，cron によって定期的に起動されるウォッチスクリプトから起動される．ウォッチスクリプトは，データベースの仮想クラスタ予約テーブルを監視し，現在の時刻においてなされていなければならないことを検出する．予約開始時刻が過ぎているにもかかわらず，インストールが開始されていない仮想クラスタがテーブル上にあれば，インストールを開始し，仮想クラスタ予約テーブルを更新してインストール中であるとマークする．逆に，予約終了時刻が過ぎていて，まだ実行中の仮想クラスタがあれば，仮想クラスタの破棄プロセスを起動し，同様にテーブルに破棄中であることをマークする．

このように，独自のデーモンプロセスを持たず，複数のスクリプトがデータベースを中心に連携する構造をとることによって，頑健性を得ている．

同様に，ゲイトウェイノード，物理計算ノード，ストレージノード上でも独自のデーモンは使用していない．各ノード上には，仮想計算機や iSCSI ターゲットを制御，監視するためのスクリプトが容易されており，これをクラスタマネージャから ssh 経由で起動することですべての動作が実現されている．さらに，これらのノードは基本的に状態を記憶していない．すべての状態は，クラスタマネージャのデータベースに一括して保持されている．このような構造をとることによって，各ノードが持つ情報に矛盾が生じる可能性を未然に防ぐとともに，頑健性を得ている．ssh の認証に用いる鍵は Rocks が物理クラスタインストール時に自動的に生成，配布するため管理者が設定する必要はない．

4.3 仮想クラスタインストールのステップ

仮想クラスタのインストールは，1) プライベートネットワーク上に VLAN を設定し，仮想クラスタのネットワークを切り分ける．2) 仮想フロントエンドをゲイトウェイノード上にインストール，3) 仮想計算ノードを起動，仮想フロントエンドからインストール，の 3 つの段階で行われる．

4.3.1 VLAN の設定

まずクラスタマネージャは，仮想クラスタで利用する仮想計算ノードの実計算ノードへの配置，およびストレージボリュームのストレージノードへの配置を決定する．次に，各仮想クラスタごとに VLAN タグを割り当てる．VLAN タグとして使用できる値は有限であるため，資源として管理して再利用する必要がある．割り当てた VLAN タグを用いて，利用する実計算ノード，ストレージノード上にタグつきのインタフェースを作成する．

4.3.2 仮想フロントエンド起動とインストール

仮想フロントエンドはゲイトウェイノード上に構築する．仮想フロントエンドのインストールは，Rocks の自動フロントエンドインストール機能を用いて行う．これは，サイトの状態と必要な Roll とそのダウンロード元を記述したファイルを用意して ISO イメージに組み込むことで，ユーザの手を煩わせることなくフロントエンドノードをインストールする機能である．ISO には，クラスタの名前，IP アドレス，ルートパスワードなどを含めることができる．

クラスタマネージャは，各仮想クラスタに対して個別の ISO イメージを生成し，その ISO イメージをブートディスクとして用いてゲイトウェイノード上で仮想計算機を起動することで仮想フロントエンドを構築する．この際，仮想フロントエンドのファイルシステムはストレージノード上の iSCSI ボリュームを用いる（図 5 B）．この際の動作は次項で詳述する．

インストールする Roll はクラスタマネージャ上に配置しておく．仮想フロントエンドは，これをネットワーク経由でダウンロードし，インストールする．この際，ネットワークとしては，パブリックネットワークが使用される．

仮想フロントエンドの構築時には，仮想クラスタ固有の設定も行わなければならない．すなわち，仮想計算ノードの MAC アドレスと IP アドレスの対応や，アプライアンスタイプの指定などである．これらの情報は，状態設定用の Roll として予約時にパッケージングしておき，Roll 名を仮想フロントエンドインストールに用いる ISO ファイルに組み込まれる設定ファイルに記述しておく．こうすることによって，これらの情報がインストール時に仮想フロントエンド内の Rocks データベースに取り込まれ，仮想計算ノードのインストール時に用いられるようになる．

仮想フロントエンドを構築するゲイトウェイノードの選択は，ラウンドロビンで行う．予約テーブルを参照して，予約時間帯において要求される空きメモリ量を持つノード群を選び，そのなかから，その時間帯内ですでに割り当てられた仮想フロントエンドの数が最も少ないノードを選択する．この条件を満たすノードが複数ある場合には，ノード番号の若い順に用いる．使用するストレージノードの選択も同様で，要求されたストレージ容量を提供できるノードのなかからラウンドロビンで選択する．

4.3.3 仮想ノードの起動とインストール

仮想計算ノードは物理計算ノード上に構築される（図 5 C）．この際，ネットワークとして仮想クラスタに割り当てられた VLAN が指定される．起動のシーケンスは Rocks が提供する機能を利用した PXE によるブートとなる．プライベートネットワーク上にはクラスタマネージャと仮想フロントエンドの双方が Rocks のフロントエンド機能を提供する形となるが，仮想計算ノードは仮想クラスタに割り当てられたタグつきのネットワークにしかアクセスができないため，唯一アクセスできる仮想フロントエンドをフロントエンドとしてインストールが行われることになる．

仮想計算ノードは仮想フロントエンドと同様に，ストレージノード上の iSCSI ボリュームをファイルシステムとして利用する．この機能は，3.3.3 項で述べたように，物理計算ノードがアタッチした iSCSI ボリュームを，VMware Server が物理的なボリュームと見なすことによって実現されている．仮想計算ノードの起動の際には，まずクラスタマネージャはストレージノードに対して，iSCSI ボリュームの割当てを依頼し，続いて物理計算ノードに対して，iSCSI ボリュームの ID を渡して仮想計算ノードの起動を依頼する．物理計算ノードは指定されたボリュームをアタッチし，そのデバイスを指定して，VMware のコンフィギュレーションファイルを生成，仮想計算ノードを起動する．

仮想計算ノードを構築する物理計算ノードの選択，ボリュームを確保するストレージノードの選択は，仮想フロントエンドの場合と同様に，要件を満たしたもののなかから，ラウンドロビンで行う．

4.3.4 複数の仮想クラスタ

単一の物理クラスタの上に複数の仮想クラスタを配備することもできる．図 5 D にこの様子を示す．上段と下段に 2 つのクラスタが構成されている．この図では，それぞれの仮想クラスタに対してゲイトウェイノードが割り当てられているが，1 つのゲイトウェイノードが複数の仮想フロントエンドをホストとすることも，ネットワークとメモリの許す限り，可能である．

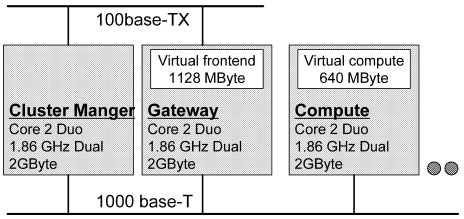


図 6 測定環境
Fig. 6 Environment for measurement.

表 1 仮想フロントエンド構築（秒）
Table 1 Virtual frontend construction time (sec.).

動作	所要時間
ISO イメージ作成	42
リストア Roll 作成	2
Rocks 再構成	155
インストール	1,632
計	1,822

4.3.5 仮想クラスタの破棄

予約終了時刻がくると，クラスタマネージャは構築した仮想クラスタを破棄し，使用していた資源を解放する．仮想フロントエンド，仮想計算ノードは仮想計算機プロセスを停止することで強制的に停止する．さらに，仮想フロントエンド，仮想計算ノードが使用していたストレージノード上のロジカルボリュームも破棄する．これによってこの仮想クラスタが使用していた領域を他の仮想クラスタに再割当てすることが可能になる．また，この仮想クラスタが利用していた VLAN タグも解放され，将来他の仮想クラスタを構築する際に再利用される．

予約終了時刻以降に仮想クラスタのストレージにアクセスする方法はない．したがって，サービスプロバイダは終了時刻以前にすべての情報を仮想クラスタから回収しておく必要がある．

4.4 仮想クラスタインストール時間の計測

仮想クラスタの構成にかかる時間を計測した．測定に用いた環境を**図 6** に示す．

まず，仮想フロントエンドの構築時間とその内訳を**表 1** に示す．表中の「Rocks 再構成」は，クラスタマネージャ内で，仮想フロントエンドに必要な Roll を整理し，配布用のファイルを生成するために要した時間である．最も時間を要したのはインストールで，27 分程度の時間を要した．この理由の 1 つは，仮想フロントエンドのインストールが，1000 base-T のプライベートネットワークではなく，100 base-TX のパブリックネットワークで行われているために，パッケー

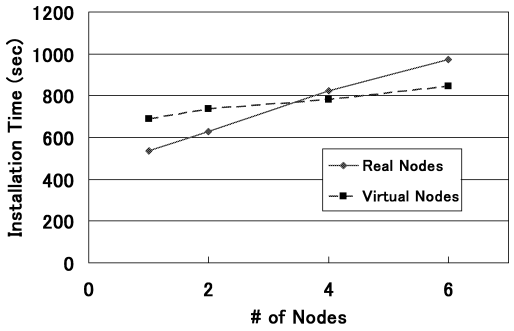


図 7 仮想ノードインストール時間
Fig. 7 Time spent to install nodes.

ジのダウンロードが低速であったためである．

この仮想フロントエンドから，仮想計算ノードをインストールした際の経過時間を**図 7** に示す．なお，この際ストレージはローカルファイルシステム上に構築した．横軸に同時にインストールした台数，縦軸に経過時間を秒で示している．仮想計算ノード上には最も基本的な Roll のみをインストールした．参考のため，物理ノードのインストールの時間も同じグラフに示している．著者の経験上，Rocks のノードのインストールにかかる時間は，とくにメモリ量が少ない領域において，メモリ量の影響を大きく受けることが分かっている．今回の実験では，実計算ノード，仮想計算ノードとも十分なメモリを割り当てているため，メモリ量による影響はない．

図 7 から，物理ノードのインストール時間と，仮想計算ノードのインストール時間は同程度であることが分かる．1 台のみインストールする場合で比較すると，実計算ノードのインストールのほうが若干遅い．これは，実計算ノードのインストールの際には VMware Server を含んだ Roll がインストールされるため，ダウンロードにかかる時間が大きくなるためである．

一方，台数を増やした際の，インストール時間の増加傾向は，仮想計算ノードのほうが大きい．これは，フロントエンドからインストールするデータをダウンロードする際にかかる時間によるものと思われる．実フロントエンドのディスクアクセスバンド幅と比較して，仮想フロントエンドのディスクアクセスバンド幅が小さいため，同時にインストールする台数の影響を受けやすいためである．

4.5 管理コストに関する考察

本システムは，クラスタプロバイダがサービスプロバイダに資源を提供する際の管理コストを低減させることを目指している．このような想定で，通常の物理的なクラスタを用いる場合と，本システムを用いた場合を，管理コストの側面で比較してみる．

通常の物理クラスタで予約リクエストに基づいてクラスタを提供するには，各リクエストに対して，1) リクエストを満たすだけの計算機を計算機プールから集め，2) それぞれの計算機に，リクエストに応じた量のメモリ，ストレージを実装し，3) ネットワークを設定し，4) オペレーティングシステムをインストールし，5) ユーザの指定したアプリケーションをインストール，設定することが必要となる．

これに対して，本システムを用いる場合には，物理クラスタ構築時に一度本システムをインストールするだけで，上記の過程がすべて仮想化，自動化される．オペレータの手を煩わせることなく，仮想クラスタを予約時刻にサービスプロバイダに提供することができる．さらに，本システムのインストール自体も Rocks で行われるため，非常に容易で，クラスタマネージャを DVD-ROM から起動してインストールし，あとは，各ノードを順番にネットワークブートしていくだけで完了する．運用中の資源の追加も容易で，ノードをネットワークに接続して，ネットワークブートするだけでよい．

このように，本システムを用いることで，クラスタプロバイダにとっての管理コストを飛躍的に低減させることができる．

5. 関 連 研 究

5.1 ORE Grid

ORE Grid [10] は，Globus Toolkit のジョブ起動機構である GRAM と連動したシステムで，ユーザに指定された環境を仮想計算機上に構築し，ユーザの指定したジョブを実行するシステムである．仮想計算機の動的構成には，クラスタインストールツール Lucie [11] を用いている．また，文献 12) では，ゴールデンイメージのキャッシュを用いることによってクラスタインストールの高速化を行っている．我々のシステムがクラスタを数日〜数週間の単位でリースすることを前提としているのに対し，ORE Grid では比較的細粒度のジョブを対象としている点が異なる．

5.2 Virtual Workspace

Virtual Workspace [13),14)] は，Globus Project [15)] の一環として行われているプロジェクトで，ジョブの実行環境を仮想計算機上に構築することを目的としている．WSRF を用いたインタフェースで仮想計算機が構成する仮想的なワークスペースを作成し，そこでユーザのジョブを実行する．

Virtual Workspace も，基本的に個々のジョブの実

行を指向している点が我々のシステムと異なる．

5.3 OSCAR による Xen クラスタインストール

文献 16) では，複数の Linux ディストリビューションに対応している点を特徴とするクラスタインストーラ OSCAR [17] を用いて，仮想計算機システム Xen を用いた仮想的なクラスタを構築している．しかし，この研究では単に OSCAR を用いたインストールの実現を示しているだけであり，本システムのように予約機構と連動した自動インストールを実現してはいない．また，ストレージとネットワークの仮想化の実現も行っていない．

5.4 Infiniband を用いたネットワークとストレージの仮想化

Cisco 社の VFrame [18] は Infiniband ネットワークと SAN を用いてネットワークとストレージの仮想化を実現する．このシステムは非常に高機能で，I/O バンド幅の制限なども可能であるが，反面非常に高価で小規模な計算機センタが導入できるものではない．

6. おわりに

クラスタインストールシステム Rocks を用いた仮想クラスタ管理システムを提案した．本システムは，VMware Server による計算機資源の仮想化，iSCSI を用いたストレージ資源の仮想化，VLAN を用いたネットワーク資源の仮想化により，効率的で安全な仮想クラスタの運用を実現する．

本システムはいまだプロトタイプ実装の段階であり，多くの課題をかかえている．

- インストール時間の詳細な計測と速度向上
  本システムでは，仮想クラスタの提供を予約ベースで行うため，仮想クラスタのインストールにかかる時間はあまり問題にはならない．とはいえ，今回の測定結果は我々にとっても意外なほど低速であった．所要時間のより詳細な内訳の解析を行い，高速化を図るとともに，仮想クラスタのインストールをバックグラウンドで行うなど，インストール時間を顕在化させない方法を検討する必要がある．
- Xen への対応
  現在，本システムは VMware Server にのみ対応しており，広く利用されている仮想計算機機構である Xen [19] を利用することはできない．これは，現在の Rocks がベースとする CentOS 4 系のインストーラが Xen に対応していないためである．2007 年春に予定されている CentOS 5 の Xen への対応後，我々も Xen に対応する予定である．
- ゲイトウェイノード，物理計算ノード，ストレージノード選択の最適化
  4.3.2 項に述べたとおり，現在の実装は，仮想クラスタを構築する際に用いる物理的なノード群の選択を簡単なラウンドロビンで行っている．このため，場合によっては，負荷の不均衡が起こる可能性がある．より高度なスケジューリングアルゴリズムを用いて，物理的なノード群の選択を最適化する必要がある．
- より高度な仮想ストレージ管理
  本システムでは，現在共有ファイルシステムとして仮想フロントエンドがマウントした iSCSI target を NFS で共有する．この構造では，共有ディスクへのアクセスが必ず仮想フロントエンドを経由することになりボトルネックとなる可能性がある．GFS [20]，PVFS2 [21]，Lustre [22]，Gfarm [23] などのクラスタファイルシステムを利用することで iSCSI ディスクへのアクセスを分散することでこれに対処する．
  また，科学技術計算の一部では，非常に高速な一時ファイルを必要とするものがある．GFS，PVFS2，Lustre などのストライピング機能を用い，高速な一時ファイルへの要請を満たすことも今後の課題である．
- 他のディストリビューション，OS への対応
  本システムは，仮想クラスタのインストールに Rocks を用いることから，Rocks の制約によって，仮想クラスタのオペレーティングシステムとして CentOS しか使用することができない．サービスプロバイダによっては，他のディストリビューション，OS を要請する可能性があることから，対応を検討する必要がある．他 OS のなかでは，圧倒的なユーザベースを持つ Windows への対応を考慮する必要があるだろう．クラスタインストール機能を持つ Windows Computer Cluser Server 2003 への対応を検討する．

謝辞 Rocks に関してご教示いただいた，SDSC Rocks チームの Mason Katz，Greg Bruno，Anoop Rajendra に感謝する．

参考文献

1) 中田秀基，横井　威，関口智嗣：Rocks を用いた仮想クラスタ構築システム，情報処理学会 HPC 研究会 2006-HPC-106 (2006).
2) VMWare. http://www.vmware.com
3) iSCSI Specification.

http://www.ietf.org/rfc/rfc3720.txt
4) 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).
5) Rocks. http://www.rocksclusters.org/
6) Sacerdoti, F.D., Katz, M.J. and Papadopoulos, P.M.: 411 on Scalable Password Service, *IEEE High Performance Distributed Computing Conference* (2005).
7) Ganglia. http://ganglia.sourceforge.net/
8) Condor. http://www.cs.wisc.edu/condor/
9) 神坂紀久子，山口実靖，小口正人，喜連川優：iSCSI を用いた PC クラスタにおけるバックエンドネットワーク統合による性能への影響評価，電子情報通信学会技術研究報告，コンピュータシステム研究会 (2006).
10) 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).
11) 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，先進的計算基盤システムシンポジウム SACSIS2003 論文集，pp.365–372 (2003).
12) 西村豪生，中田秀基，松岡聡：仮想計算機と仮想ネットワークを用いた仮想クラスタの構築，情報処理学会研究報告，2006-HPC-107, pp.73–78 (2006).
13) Virtual WorkSpace. http://workspace.globus.org/
14) 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).
15) Globus Project. http://www.globus.org
16) Vallée, G. and Scott, S.: Xen-OSCAR for Cluster Virtualization, *Workshop on XEN in HPC Cluster and Grid Computing Environments* (*XHPC '06*) (2006).
17) OSCAR: open source cluster application resources. http://oscar.openclustergroup.org/
18) Cisco VFrame Server Fabric Virtualization Software. http://cisco.com/en/US/products/ps6429/products_data_sheet0900aecd8029fc58.html
19) Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauery, R., Pratt, I. and Warfield, A.: Xen and the Art of Virtualization, *SOSP 2003* (2003).
20) Red Hat Global File System. http://www.redhat.com/software/rha/gfs/
21) PVFS2. http://www.pvfs.org/
22) lustre. http://www.lustre.org/
23) Tatebe, O., Morita, Y., Matsuoka, S., Soda, N. and Sekiguchi, S.: Grid Datafarm Architecture for Petascale Data Intensive Computing, *Proc. 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid* (*CCGrid 2002*), pp.102–110 (2002).

(平成 19 年 1 月 22 日受付)
(平成 19 年 5 月 1 日採録)

中田 秀基（正会員）

昭和 42 年生．平成 2 年東京大学工学部精密機械工学科卒業．平成 7 年同大学大学院工学系研究科情報工学専攻博士課程修了．博士（工学）．同年電子技術総合研究所研究官．平成 13 年独立行政法人産業技術総合研究所に改組．現在同所グリッド研究センター主任研究官．平成 13 年より平成 17 年度まで東京工業大学客員助教授を兼務．グローバルコンピューティング，並列実行環境に関する研究に従事．

横井 威（正会員）

昭和 50 年生．平成 12 年旧東京商船大学流通情報工学課程卒業．平成 12 年〜平成 17 年ジャパン石油開発株式会社勤務．平成 18 年〜現在，独立行政法人産業技術総合研究所グリッド研究センター勤務，研究員．

江原 忠士（正会員）

昭和 45 年生．平成 6 年電気通信大学電気通信学部電子物性工学科卒業．平成 8 年同大学大学院電気通信学研究科電子物性工学専攻修士課程修了．修士（工学）．同年株式会社数理技研入社．平成 19 年株式会社数理先端技術研究所入社．同年株式会社数理技研退職．現在株式会社数理先端技術研究所取締役．

谷村　勇輔（正会員）

昭和 51 年生．平成 11 年同志社大学工学部知識工学科卒業．平成 16 年同大学大学院工学研究科知識工学専攻博士課程修了．同年独立行政法人産業技術総合研究所グリッド研究センター特別研究員．平成 17 年同センター研究員．博士（工学）．最適化手法の実装等の並列応用計算，グリッドコンピューティングの基盤システムに関する研究に従事．HPCS2005 論文賞．

小川　宏高（正会員）

1971 年生．東京大学大学院工学系研究科博士課程中退．1998 年より東京工業大学大学院情報理工学研究科助手．2003 年より産業技術総合研究所グリッド研究センター研究員．主な研究は，自己反映計算に基づいた拡張可能な Java 言語向け Just-In-Time コンパイラ OpenJIT，グリッド技術を用いてアプリケーションの実行をユーティリティサービスとして提供する GridASP 等．プログラミング言語処理系，仮想化技術，並列処理，グリッド技術等に興味を持つ．

関口　智嗣（正会員）

昭和 34 年生．昭和 57 年東京大学理学部情報科学科卒業．昭和 59 年筑波大学大学院理工学研究科修了．同年電子技術総合研究所入所．情報アーキテクチャ部主任研究官．以来，データ駆動型スーパコンピュータ SIGMA-1 の開発等の研究に従事．平成 13 年独立行政法人産業技術総合研究所に改組．平成 14 年 1 月より同所グリッド研究センターセンター長．並列数値アルゴリズム，計算機性能評価技術，グリッドコンピューティングに興味を持つ．市村賞受賞．日本応用数理学会，ソフトウェア科学会，SIAM，IEEE 各会員．