統合ディスクレスネットワーク基盤システム

古橋貞之

i0611262@coins.tsukuba.ac.jp

筑波大学 第三学群情報学類

小規模な企業や教育機関でさえコンピュータシステムが導入されることが当然となっているが、ネットワークの構築には高度な知識と多大な労力が必要とされる。多数のコンピュータを統一的に扱えるようにする基盤システムを開発することで、これらのコストを軽減することができる。本稿で述べるシステムは、1台のコンピュータをCDなどのメディアから起動し、続いて他のコンピュータを次々にネットワークブートする。そしてその上にプラグイン型のアーキテクチャを使用してネットワークサービスを構築する。

Foundation System for Unified Diskless Network

FURUHASHI Sadayuki

College of Information Sciences, University of Tsukuba

While it becomes common that even small offices or educational institutions introduce computer system, advanced knowledge and a great deal of labor are needed to construct a computer network. These costs can be reduced by developing foundation system that unifies many computers. This paper presents following system; Boot one computer from a removable media such as CD, and other computers one after another using network boot. Then compose network services using plug-in model architecture.

1 はじめに

情報化社会の進化に伴い、コンピュータが我々の生活に必要不可欠なものとなって久しい。学校教育においてもコンピュータは一般化しており、高等学校教育においては一般教科「情報」が開始され、小学校においては「総合的な学習の時間」における調べ学習などでコンピュータシステムの活用が推進されている。大学においても、2007年度からと試算される大学全入時代を前に各大学が差別化を図る中で、情報教育環境に関する施策も行われている。しかしハードウェアの急激な進化と複雑化に伴い、コンピュータネットワークの構築には高度な技術力と知識が必要とされるようになり、その構築にかかる時間的、労力的コストは増大している。小規模な機関にとって、このようなコストは大きな負担となる。

だが見方を変えれば、多数のコンピュータが高速・低遅延のネットワークに接続された状態で同時に稼働するシステムは、それ自体有用な資源である。一般的なPCを多数接続したPCクラスタは、PCの飛躍的な性能向上と低価格化、そして高速ネットワークの普及によって、様々な研究分野における大規模計算に利用され始めている[1)]。しかし多数のコンピュータを正しく設定し、適切に動作するネットワークを構築するには、高度な知識と技術に加えて時間と忍耐が必要とされる。

以上のような現状に対して、私はコンピュータネットワークをいかに簡単に構築し、その資源をうまく利用するかという点に着目した。ネットワークの構築が困難であることの原因は、多数のコンピュータの一括して扱うことができず、互いに独立した多数のコンピュータに対して同じような処理を何度も施さなければならない点にある。そこで本稿では、多数のコンピュータを統一的に扱えるようにする基盤を構築することで、極めて簡単な手順でコンピュータシステムを構築できるようにするシステムを提案する。

2 従来手法

前章で挙げた問題点に対する従来の解決手法として、Live CD[2)3)4)] とシンクライアントシステム[5)6)] がある。

Live CD は CD や DVD などのメディアから起動する OS であり、手軽に同じ環境のコンピュータを多数用意することができる。Live CD の手軽さは、以下の点によって特徴づけられる。

- インストールが不要
- HDD レスで動作する
- ハードウェアの構成に依存しない
- 初期費用が発生しないか、安価である

シンクライアントシステムは、サーバーですべてのアプリケーションやファイルを管理し、クライアントには最小限の機能しか持たせないようにするシステムである。サーバーからクライアントを一元管理できるため、システムの管理が容易になる。しかしサーバーに高い性能が必要になるため、初期費用が大きくなるという点が問題となる。

本システムは、Live CD と同等の手軽さを持ちつつ、シンクライアントのようにクライアントコンピュータの管理を容易にすることを目指す。

3 提案手法

3.1 概案

本システムは、まず CD や DVD、あるいは USB メモリなどのメディア（以下、ブートメディアと呼ぶ）からコンピュータを 1 台起動する。続いてそのコンピュータをサーバーとして、任意の台数のクライアントコンピュータをネットワークブートする。さらにこのコンピュータ群に対して、プラグイン型のアーキテクチャを使用して、協調したネットワークを構築する。協調したネットワークとは、そのネットワークの目的に応じて、ファイルの共有システムや、パスワードの一元管理システム、あるいは並列計算クラスタのミドルウェアなどが、適切に動作するネットワークを指す。OS には Linux を使用する。

以下、ブートメディアやネットワークからコンピュータを起動する手法を 3.2 節で、協調したネットワークの構築する手法を 3.3 節で述べる。

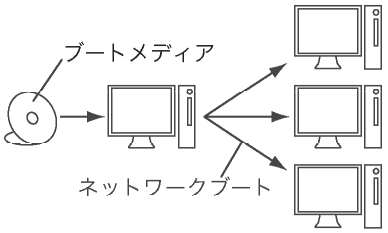

図 1 ディスクレスブート
Fig. 1 Diskless boot

3.2 ディスクレスブート

ディスクレスブートの構造を上図（図 1）に示す。

起動する際、Live CD と同等の手軽さを得るために、ハードウェアを自動検出を行う。これによってハードウェアの構成に依存せずに動作できる。また、データの書き込みは RAM（RAM ディスク）や、USB メモリなどの外部ストレージに行う。これによって HDD レスで動作し、既存の環境を壊さずに利用できる。

クライアントコンピュータを動作させる方法としては、画面転送方式とネットワークブート方式が考えられる。画面転送方式は、すべてのアプリケーションをサーバーで実行し、スクリーン、キーボード、マウスなどの入出力だけをサーバーとクライアントの間でやりとりする方式である。この方式はインターネットレベルのネットワーク帯域があれば使用することができるが、高性能なサーバーコンピュータが必要になる点が問題となる。ネットワークブート方式は、サーバーが提供するディスクイメージを使ってクライアントを起動する方式である。この方式はクライアントコンピュータの処理能力を活かすことができるが、高速なネットワークが必要であり、LAN での利用に限定される。本システムは十分なネットワーク帯域が得られる LAN 環境を前提とし、ネットワークブート方式を用いる。

3.2.1 initrd

Linux カーネルは、起動時に initrd と呼ばれるディスクイメージを使用する機能を持つ。本システムはこれを拡張することでディスクレスブートを実現する。initrd は次のように使用される；Linux に対応したブートローダは、Linux カーネルと共に initrd をロー

ドし、Linux カーネルに処理を移す。Linux カーネルは初期化を行った後、initrd を RAM ディスク上に展開し、これをルートファイルシステムとしてマウントする。続いてルートディレクトリ（すなわち initrd）に保存された linuxrc という名前のファイルを実行する。linuxrc はディスクドライブやファイルシステムなどのドライバをロードし、実際のルートファイルシステムをマウントする。そしてルートファイルシステムを initrd から実際のルートファイルシステムに入れ替える。ここで linuxrc の処理は終了し、Linux カーネルは /sbin/init ファイルを実行する。linuxrc によってルートファイルシステムの入れ替えが行われたので、/sbin/init は initrd 上ではなく実際のルートファイルシステム上のファイルである。

3.2.2 カーネルと initrd のロード

ブートメディアから起動する場合は、BIOS がそのメディアからの起動をサポートしていなければならない。サポートしていれば、そのメディアに保存されたブートローダに処理が移る。ブートローダはブートメディアからカーネルと initrd をロードし、カーネルに処理を移す。ブートローダには ISOLINUX[7] を使用する。

ネットワークからの起動には、PXE[8] を使用する。PXE によるブートの手順を下図（図2）に示す。

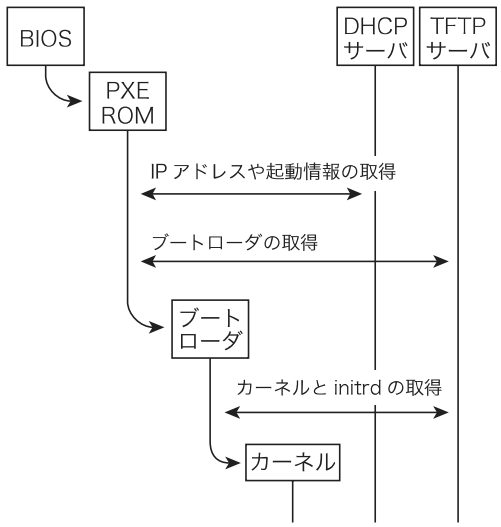

図2 PXE によるブートシーケンス
Fig. 2 Boot sequence of PXE.

PXE に対応した NIC がある場合は、BIOS から NIC に搭載された PXE ROM に処理が移る。PXE に対応した NIC が無い場合は、EtherBoot[9] を補助に使用する。PXE ROM は DHCP サーバーから自身の IP アドレスや TFTP サーバーの IP アドレスを取得し、続いて TFTP サーバーからブートローダを取得する。この DHCP サーバーと TFTP サーバーは、ブートメディアから起動したコンピュータで実行する。これは後述する協調したネットワークを構築するシステムの一部として実行する。ブートローダは TFTP サーバーからカーネルと initrd を取得し、カーネルに処理を移す。ブートローダには PXELINUX[10] を使用する。

3.2.3 ハードウェア自動検出

ブートメディアまたはネットワークから OS を起動するために必要なハードウェアは、CD/DVD ドライブや HDD などのディスクドライブと、NIC である。グラフィックカードやサウンドカードは必要ない。ディスクドライブを使用するためには、デバイスドライバのロードと、ファイルシステムタイプ（NTFS、ext3 など）の判別が必要である。NIC を使用するためには、デバイスドライバのロードと、IP アドレスの割り当てが必要である。

まずデバイスドライバをロードする。現在一般的な PC は PCI アーキテクチャを使用しているため、PCI バスに接続されるハードウェアを検出の対象とする。

PCI バスに接続されたデバイスからは、その Vendor ID、Device ID、Class ID を取得できる。一方で Linux カーネルは、ドライバ名と、そのドライバで制御できるデバイスの Vendor ID と DeviceID の組の対応表を持つ。この2つを照らし合わせることで適切ドライバを検出し、それをロードする。この段階で PCI バスに直接接続されたデバイスが使用可能になる。

続いて Class ID を参照してデバイスの種類を判別し、個別に処理を行う。現時点では、CardBus コントローラ、USB コントローラ、IEEE1394 コントローラ、SCSI コントローラ、IDE コントローラ、Serial ATA コントローラ、NIC を判別し、処理を行う。

CardBus コントローラには、PCI バスと同様に USB などの各種コントローラが接続されるため、さらに再帰的に検出を行う必要がある。検出には PCI バスと同じ方法が使用できるため、PCI バスと同じ処理を行う。

USB コントローラには多彩なデバイスが接続されるが、PCI バスと同じ方法が使用できるため、PCI バスと同じ方法を用いる。ただし Class ID を用いた再帰的な検出は行わない。Class ID からストレージが接続されていることが判別された場合は、Vendor ID と Device ID に関わらず USB ストレージ用のドライバもロードする。

SCSI、IDE、Serial ATA コントローラにはディスクドライブが接続されることがほとんどである。IEEE1394 には DV 機器（ビデオカメラなど）が接続されることも多いが、これは起動時には必要ない。よって、これらのコントローラを検出した場合は、常にそれぞれのディスクドライブ用のドライバをロードする。

ディスクドライブを検出した場合、ファイルシステムタイプの判別を行う。これは考えられるすべてのファイルシステムタイプでマウントを試みることで行う。この際、パーティションが存在する場合（すなわち HDD や USB メモリなどの場合）は NTFS → FAT → ext3 の順から試行し、存在しない場合（すなわち CD や DVD などの場合）は ISO9660 → UDF → HFS+ の順から試行することで、より早くファイルシステムタイプを判別できるようにする。

NIC を検出した場合、IP アドレスを割り当てる。まず DHCP による自動割り当てを試みる。失敗した場合は、Auto-IP[11] を使用して割り当てる。Auto-IP は、169.254.0.0/16 のアドレスレンジの中から適当に IP アドレスを 1 つ選び、そのアドレスが既に使用されているか ARP を利用して確かめるという試行を繰り返し、使用されていない IP アドレスを探し出すものである。DHCP クライアントと Auto-IP の実装には、共に BusyBox[12] を使用する。

3.2.4 ディスクの共有

ブートメディアから起動するコンピュータは、メディア上の圧縮されたディスクイメージをルートファイルシステムとしてマウントして起動する。また、この圧縮ディスクイメージをネットワークを通じてクライアントと共有する。クライアントはこれをマウントして起動する。この構造を上図（図3）に示す。

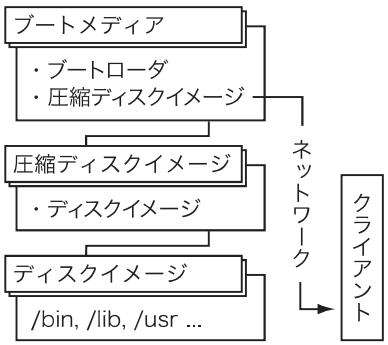

図3 圧縮ディスクイメージの共有
Fig. 3 Sharing of compressed disk image.

圧縮ディスクイメージを共有する際、データを伸張せずにそのまま共有する。これによりネットワークには圧縮されたデータが流れるため、トラフィックを削減できる。また、元々圧縮されているデータをそのまま共有するため、サーバーに圧縮の負荷はかからない。ディスクイメージの圧縮には Squashfs[13] を使用し、共有には NBD[14] を使用する。Squashfs は読み取り専用の圧縮ファイルシステムであり、データは読み取り時に透過的に伸張される。NBD は Linux カーネルの機能であり、リモートコンピュータ上のファイルやデバイスを、ローカルのブロックデバイスとして認識させる。

3.2.5 Copy-on-Write

データの書き込みは、HDD 以外の場所に書き込み可能な領域を確保し、そこに元のディスクイメージに対する差分を書き込むことで行う (Copy-on-Write)。これにより、ブートメディアに CD や DVD などの読み取り専用メディアを使用することができる。また、ブートメディアから起動するコンピュータとネットワークブートするコンピュータ群で、一つのディスクイメージを共有できる。差分データを書き込む領域には基本的に RAM（RAM ディスク）を使用するが、ユーザーからの指定があった場合は、USB メモリや HDD な

どの Linux カーネルで使用可能なストレージを使用する。RAM ディスクに差分データを保存する場合は、シャットダウン時に差分データを抹消することができる。別のストレージを使用した場合は、シャットダウン後も差分データを保持することができる。

差分データを書き込むストレージを指定するには、そのストレージ上にあるファイルの名前を1つ指定する。ファイルシステムタイプはハードウェアの自動検出時に判別するため、ユーザーが指定する必要はない。検出したストレージの中から、指定されたファイルを探索する。ファイルシステムタイプやパーティション番号を指定する必要がないため、分かりやすい。

差分データを書き込む領域には、NTFS でフォーマットされたパーティションも使用できる。現時点で Linux カーネルは NTFS に対して、新しいファイルを作成せず、かつファイルサイズを変更しない書き込みしかできない[15]。しかし NTFS を直接使用せず、あらかじめユーザーに大きめのファイルを作成しておいてもらい、これをループバックマウントして使用することで、NTFS 上のファイルサイズを変更せずにすべての操作が可能になる。これによって Windows がインストールされている環境でも HDD に差分データを保存することができる。

Copy-on-Write の実装には cowloop[16] を使用する。

3.3 協調したネットワークの構築

起動したコンピュータ群に対して、ネットワーク中での役割（ロール）を割り当てる。これによってサーバーやクライアントなどの役割分担をさせ、ファイル共有システムやパスワードの一元管理システムなどのネットワークサービスを構築する。構築するサービスはプラグインとして扱い、1つのロールに対して複数のプラグインを割り当てる。このアーキテクチャを次の図（図4）に例示する。

2.3.1 ロールの適用

ロールには「プリセットロール」と「オプショナルロール」がある。プリセットロールには3種類あり、「すべてのコンピュータに適用するロール」「ブートメディアから起動したコンピュータに適用するロール」「ネットワークブートしたコンピュータに適用するロール」である。オプショナルロールは、ユーザーが自由に割り当てられるロールである。この割り当てはブートパラメータで行い、ブートパラメータはブートローダで指定する。ブートメディアから起動する場合は、あらかじめブートメディアに設定しておくか、起動時にキーボードから入力する。ネットワークブートする場合は、サーバーからクライアントの MAC アドレスに基づいて制御するか、起動時にキーボードから入力する。

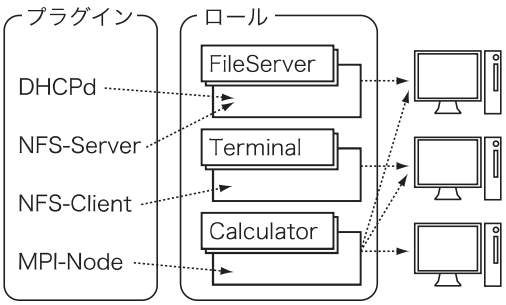

図4 ネットワークの構築
Fig. 4 Construction of the network.

2.3.2 プラグインの構造

ディスクレスブートの処理が終了した後、プラグインを適用していく。プラグインは実行ファイルとメタデータを含む1つのディレクトリであり、プラグインを適用するとは、その実行ファイルを実行することを指す。下図（図5）にプラグインのディレクトリ構造を示す。

実行ファイルのために他のデータが必要な場合は、すべてリソースディレクトリに入れることとし、ディレクトリ構造のトップレベルには図に示すファイルとディレクトリ以外を置いてはならないことにする。これにより

```
ExamplePlugin.bdl/
├─ start          … 実行ファイル
├─ info.xml       … メタデータ
├─ default.conf … デフォルト設定ファイル
└─ resources/   … リソースディレクトリ
```

図5 プラグインバンドル
Fig. 5 Plugin bundle.

プラグインの全容を把握しやすくする。メタデータにはプラグイン名、バージョン、適用順制御情報、依存関係情報を記述する。

3.3.3 プラグインの適用

適用順制御情報には、そのプラグインより前に適用されなければならないプラグインと、後に適用されなければならないプラグインを記述する。プラグインを適用する際、適用するプラグインすべての適用順制御情報を読み取り、関係を満たすように適用していく。関係が循環している場合は、循環しているプラグインを取り除いて、エラーメッセージを出力する。相互に関係がないプラグインは並列して適用し、起動時間を短縮する。

3.3.4 プラグインの管理

依存関係情報には、そのプラグインを適用するために必要な他のプラグインやファイルを記述する。適用順には影響しない。プラグインのインストール時にはこの情報を使用し、プラグインを適用できる環境が整っているかを検査する。

4 ディストリビューション非依存

本システムは Linux カーネルを使用するが、特定の Linux ディストリビューションに依存しない。これは initrd 内で動作が完了し、その後は HDD から起動する場合と同じように動作できるためである。以下のディストリビューションが動作することを確認した。

- CentOS 4
- Fedora Core 5/6
- Mandriva Linux 2006/2007
- Momonga Linux 3
- openSUSE 10.1
- Progeny Debian 3.0 PR2
- Ubuntu Linux 6.06
- Vine Linux 3.2

これによって、利用環境に合わせて Linux システム全体を柔軟に構築することができる。

5 評価

本システムはクライアントを動作させるためにネットワークブート方式を用いるが、この方式は高速なネットワークが必要であるとを前述した。そこでクライアントを起動する際にどの程度のトラフィックが発生するかを測定した。ネットワーク帯域として 1000BASE-T の環境と 100BASE-TX の環境の2つで試験した。

5.1 評価環境

2 台のコンピュータを L2 スイッチを経由して Ethernet で接続し、一方をブートメディアから起動した。これをサーバーと呼び、他方をクライアントと呼ぶ。この試験では圧縮ディスクイメージを RAM ディスク上に置き、これをネットワークで共有した。

圧縮ディスクイメージは CentOS 4.4[17] を使用して作成した。クライアントはランレベル 5（グラフィカルログイン）で起動し、自動的にログイン処理が行われてグラフィカルデスクトップが起動するようにした。使用した機器とソフトウェアの情報を表 1 に示す。

5.2 トラフィック

クライアントをネットワークブートし、発生したトラフィックを計測した。トラフィックはサーバー側で計測し、NIC を通過した総データ量とその時点の時刻を 1 秒間隔で記録するプログラムを使用した。クライア

コンピュータ	
型名	HP Compaq Buisness Desktop dc7600 MT/CT
CPU	Intel Pentium4 660 3.6GHz
Memory	DDR2 1GB
NIC	Broadcom BCM5752 PCI-Express
OS	CentOS 4.4 カーネル 2.6.18

ネットワーク	
L2 スイッチ	Buffalo LSW2-GT-8ESR
ケーブル	カテゴリー 6 （1000BASE-T 時） カテゴリー 5 （100BASE-TX 時）

表 1 評価環境
Table 1 Experimental environment.

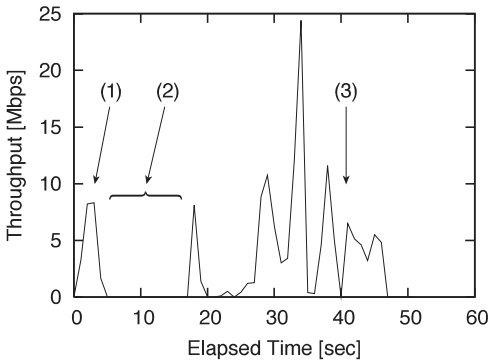

図6 ネットワークトラフィックの時間変化（1000BASE-T）
Fig. 6 Time variation of network traffic (100BASE-T).

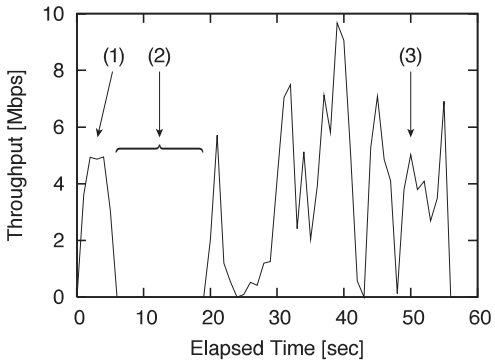

図7 ネットワークトラフィックの時間変化（100BASE-TX）
Fig. 7 Time variation of network traffic (100BASE-TX).

ント側でも何らかの処理が行われる際に記録スクリプトを動作させ、その時刻で行われている処理内容を記録した。サーバーとクライアントのハードウェアクロックは、あらかじめ正確に合わせておいた。

1000BASE-T 時の測定結果を図6に、100BASE-TX 時の測定結果を図7に示す。スループットはサーバーからクライアントへの下り方向の流量が支配的であるため、下り方向の値を示す。

計測開始直後のピーク（図中の（1））は、ブートローダによる TFTP サーバーからのカーネルと initrd の取得によるものである。その後トラフィックが発生していない時間（図中の（2））は、カーネルの初期化と、本システムのディスクレスブートのための処理が行われている。本システムの動作が開始したのは 1000BASE-T 時は7秒、100BASE-TX 時は１０秒であり、動作時間は共に 11 秒であった。

本システムの動作が終了した後、Linux ディストリビューションの起動処理が開始され、大きなトラフィックがに発生している。この時刻からグラフィカルログインが行われた時刻（図中の（3））までの間に発生したトラフィックは、1000BASE-T 時は全体の 81%、100BASE-TX 時は全体の 83% と大きな部分を占めている。またグラフィカルログインが行われてから約５秒間も大きなトラフィックが発生し続けており、起動時に発生するトラフィックがシステム設計におけるボトルネックになると考えられる。

発生したトラフィックの総量は、1000BASE-T 時は下り 155MB 上り 4MB、100BASE-TX 時は下り 155MB 上り 5MB であった。計測開始からトラフィックが収束するまでに要した総時間は、1000BASE-T 時は４６秒、100BASE-TX 時は５５秒であった。また、起動時の差分データは 4MB 程度と少なく、差分データを書き込む領域に RAM ディスクを使用している場合でも RAM への圧迫は少ない。

6 今後の計画

6.1 シングルポイント障害の回避

本稿で述べた圧縮ディスクイメージの共有方法では、ブートメディアから起動したコンピュータ１台だけをサーバーとしているため、このコンピュータに障害が発生するとすべてのクライアントがハングアップしてしまう。この問題に対しては、（1）クライアントの RAM 上に圧縮ディスクイメージ全体をコピーする、（2）サーバーを複数用意してフェイルオーバーが可能なシステムを開発する、（3）オーバーレイ的なアプローチでディスクイメージを複数のコンピュータに分散・重複保持させるという解決策がある。（1）は既に実装しているが、圧縮ディスクイメージのサイズが大きい場合は RAM への圧迫が大きくなってしまう。そこで（3）の実装計画を進めている。

6.2 ベースディスク

本システムは特定の Linux ディストリビューションに依存しないため、目的に合わせて自由にインストールしたディストリビューションを使って圧縮ディスクイメージを作成できる。しかしこれは同時に、本システムを利用するために何らかのディストリビューションをインストールしなければならないことを意味する。これは手軽さを損なっている。そこで、小規模な構成のシステムを提供し、これをベースにして圧縮ディスクイメージの作成が行えるようにすることを考えている。

参 考 文 献

1) PC Cluster Consortium. http://www.pccluster.org/
2) Klaus Knopper: Building a self-contained auto-configuring Linux system on an iso9660 filesystem, the Annual Linux Showcase (October 2000)
3) KNOPPIX Japanese edition. http://unit.aist.go.jp/itri/knoppix/
4) Berry Linux. http://yui.mine.nu/berry/
5) Citrix Presentation Server. http://citrix.com/
6) Ardence. http://www.ardence.com/
7) ISOLINUX. http://syslinux.zytor.com/iso.php
8) Intel Corporation: Preboot Execution Environment (PXE) Specification Version 2.1 (1999)
9) EtherBoot. http://www.etherboot.org/
10) PXELINUX. http://syslinux.zytor.com/pxe.php
11) Dynamic Configuration of IPv4 Link-Local Addresses, RFC3927 (2005)
12) BusyBox. http://www.busybox.net/
13) Squashfs. http://squashfs.sourceforge.net/
14) Network Block Device. http://nbd.sourceforge.net/
15) Linux-NTFS Project. http://www.linux-ntfs.org/
16) cowloop. http://www.atconsultancy.nl/cowloop/
17) CentOS. http://www.centos.org/