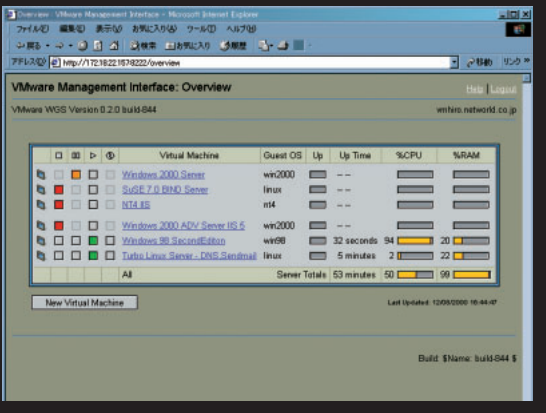
1台のサーバで複数のバーチャルマシンを実現するVMware GSX Server

ネットワールドは、米VMwareの開発したバーチャルマシンを実現するVMwareのLinuxサーバ向け製品「VMware GSX Server」とASP／ISP／iDC向け「VMware ESX Server」を12月11日より発売した。

今回、GSX Server／ESX Serverで強化された機能は、バーチャルマシン間や既存のネットワーク間でTCP/IP、Microsoft Networking、Samba、Netware、NFSをサポートしたほか、ディスク管理で最大64Gバイトのバーチャルディスクサポート、SCSI CD-ROMをバーチャルSCSIあるいはIDEとして利用可能、読み取り専用／取り消し可能ディスクモードをサポートといった点で、サーバ機で十分な性能が発揮できるようになった。

また、Webベースのリモート管理インターフェイスを備え、マシンの前に行かなくてもネットワークを通じて複数のバーチャルマシンをリモート管理できる機能を備えている。

Webサーバやメールサーバ、DNS、ファイルサーバといったネットワークを使ったサービスは、相互に影響が及ぼさないようにそれぞれ別のサーバで運用することが多い。VMwareでは、バーチャルマシン上のゲストOSがハングアップしても、ホストマシンには影響を及ぼさないので、複数のバーチャルマシン上で別々のサービスを提供することで、1台のサーバ機で複数のサービスを安全に運用することが可能になる。複数のサーバマシンを用意するよりもコストを削減できるというメリットがある。



発売日	2000年12月11日
発売	株式会社ネットワールド
TEL	03-5210-5100
価格	36万4000円
URL	http://www.networld.co.jp/

Hardware　発売日　2001年1月20日

Web、メール、DNS／DHCPアプライアンスサーバ SOOXi 2001

URL　http://www.nlcomputer.com/

発売　ノーザンライツコンピュータ株式会社
TEL　044-850-2238
価格　34万8000円〜

ノーザンライツコンピュータは、ハードウェア／ソフトウェアともに自社開発したアプライアンスサーバシリーズ「SOOXi（ソークシイ）2001」を1月20日より発売する。

Webサーバの「SOOXi 2001 Appliance Web Server」と、メールサーバの「同Mail Server」、DNS／DHCPサーバの「同DNS/DHCP Server」の3モデルが用意されている。

ハードウェアは、1UラックマウントタイプのSOOXi S100と、4UラックマウントのSOOXi S440が用意されている。S100は、PentiumIII 800M〜1GHzを2基搭載可能で、メモリ最大4Gバイト、ホットスワップ対応SCSIハードディスクを2台まで内蔵可能。2スロットの64ビット66MHz PCIスロットを備えている。S440は、PentiumIII Xeonを最大4基搭載可能で、メモリ最大16Gバイト、ホットスワップ対応SCSIハードディスクを5台まで内蔵可能となっている。



Hardware　発売日　2000年12月11日

CrusoeとMobile Linux搭載のインターネット端末 FLORA-ie 55mi

URL　http://www.hitachi.co.jp/Prod/comp/OSD/pc/ie/flora-ie.htm

発売　株式会社日立製作所
TEL　0120-8856-50
価格　オープン価格

日立製作所から、Mobile LinuxとCrusoeプロセッサ「TM3200」を搭載したインターネット情報端末「FLORA-ie 55mi」が12月11日より発売された。ペン入力によるワンタッチ操作や手書き入力機能を備えていて、主にISPやネット証券／ネット銀行などの特定サービス向けとなっている。価格は500台ロット時で9万7800円。開発評価用モデルが13万6800円。

CPUに米TransmetaのTM3200-400MHz、OSにMobile Linuxを採用し、メモリ64Mバイト（最大192Mバイト）、10.4インチTFT液晶ディスプレイ（800×600ピクセル64万色表示）を搭載。PCカードスロット（TypeII）×2、USBポート×2を装備する。また、ディスクレス設計となっており、48Mバイト（最大128Mバイト）のコンパクトフラッシュにLinuxを圧縮して格納する。サイズは280（W）×220（D）×25.8（H）mm。本体重量は約1.25kg。バッテリーは、ACアダプタのほか、Sサイズ（駆動時間が3.5時間）またはLサイズ（同7時間）のリチウムイオン電池を利用できる。



Software　発売日　2000年12月14日

Linux対応の統合ミドルウェアパッケージ IBM Small Business Suite for Linux V1.6

URL　http://www.ibm.co.jp/

発売　日本アイ・ビー・エム株式会社
TEL　0120-04-1992
価格　7万9900円

日本IBMは、Linux対応の統合ミドルウェアパッケージ「IBM Small Business Suite for Linux V1.6」を12月14日に発売した。データベース、電子メール、Webアプリケーションサーバ、専用の導入支援ツールで構成されていて、e-businessを実現するためのシステムをLinux上に構築できる。価格は7万9900円で、別途1ユーザーあたり1万4400円のクライアントライセンスが必要。

DB2 UDBワークグループ・エディションV7.1、WebSphere Application Serverスタンダード版V3.0.2、ロータスドミノfor Linux R5.0.4が含まれる。

また、zSeriesとメインフレーム用Linux for S/390で動作するLinux製品を強化し、DB2 UDB EE V7.1、同EEE V7.1（クラスタ対応）、DB2コネクト、WebSphere Application Serverスタンダード版V3.5、同アドバンスド版V3.5（zSeries対応）を発売した。



Software　発売日　2001年1月11日

5つの新機能を追加したWebグループウェア iOfficeV3

URL　http://www.neo.co.jp/ioffice/

発売　株式会社ネオジャパン
TEL　045-912-2145
価格　3万9800円（5ユーザー）〜

ネオジャパンのWebグループウェアiOffice2000がバージョンアップされ、「iOfficeV3」として1月11日に発売される。価格は5ユーザーで3万9800円から。サーバマシンには、Linux、Windows 2000／NT／95／98、UNIXなど多くのOSが利用可能。

新たに「予定通知」「備品管理」「購買予約」「レポート提出」「交通費精算」の5つの機能を追加し、全部で18の機能を提供する。画面デザインをリニューアルし、従来の機能も改良されている。

アドインモジュールによって、iモード／EZWeb／H"に対応する携帯・PHS機能、ザウルス／Windows CEなどに対応するPDA機能、Palmデバイスと通信またはデータシンク機能を利用することが可能。



Software　　発売日　2000年12月14日

Oracleデータベース監視ソフトウェア DB Monitoring Manager for ORACLE

URL　http://www.hde.co.jp/

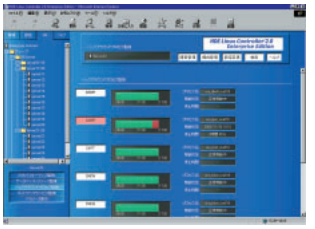
発売　株式会社ホライズン・デジタル・エンタープライズ
TEL　03-5456-3260
価格　オープンプライス

ホライズン・デジタル・エンタープライズ（HDE）は、複数台のLinuxサーバを一元管理する「HDE Linux Controller 2.0 Enterprise Edition」用Oracleデータベース監視モジュール「DB Monitoring Manager for ORACLE」を12月14日より発売した。

監視できるのは、SGAパフォーマンス、データベースリソース、バックグラウンドプロセス、ネットワークサービスといった項目で、グラフによりわかりやすく表示される。また基本情報とパフォーマンスに影響するパラメータを一覧することも可能。

複数のOracleデータベースのリソース、パフォーマンス、プロセスなどを監視することにより、リソースの枯渇や、パフォーマンスの低下、プロセスの停止を検知して、管理者宛てに警告メールなどで通知することが可能。また、バックグラウンドプロセスが停止した場合に、インスタンスの再起動を行うこともできる。

Miracle Linux＋Oracle8i for Linuxで構築された、Webとデータベースとの連携システムの一元管理が可能になる。



Software　　発売日　2000年12月13日

Linux対応BtoB ECパッケージ B@rrierFree ORACLE版

URL　http://www.e-barrierfree.com/

発売　楢崎産業株式会社
TEL　03-3363-0452
価格　68万円

楢崎産業は、Linuxに対応した企業間（BtoB）EC専門ソフト「B@rrierFree ORACLE版」を12月13日より発売した。すでに販売しているPostgreSQLを使用したB@rrierFreeの上位版。

顧客管理、受発注／見積もり、商品管理、アドレス帳、自動メール通知といった機能を備えており、Web上でリアルタイムに見積書を作成、買い物カゴ／アドレス帳を利用した発注、4段階の取引先ランク管理が行える。

B@rrierFreeはソースコードを公開しているため、ユーザーが自由なカスタマイズを行える。画面デザイン、機能追加や社内システムとの連携などにも対応が可能。iモードページを自動生成する機能をオプションで提供している。



Software　　発売日　2000年12月24日

Linuxのext2に対応したデータ復元ソフト FINALDATA Linux

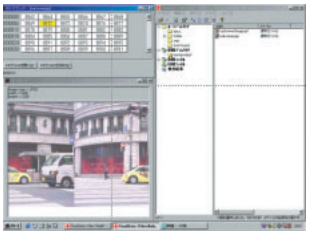
URL　http://www.alphaomega.co.jp/

発売　株式会社アルファ・オメガソフト
TEL　03-5575-2201
価格　オープンプライス

アルファ・オメガソフトは、Linuxのext2に対応するデータ復元ソフト「FINALDATA Linux」を12月24日に発売した。Windows版FINALDATAをSolarisとLinuxに対応させたもので、操作ミスで誤って削除したり、fdiskでフォーマットしたデータ、セクタ異常でハードディスクの一部が読み出せなくなった場合に、残っている情報を元にファイルを復元することができる。

実際の復元作業はWindowsマシンで行う。ハードディスクを取り外してWindowsマシンに接続するか、対象となるシステムが動作している場合には、Linux上にインストールしたプログラムからネットワーク経由でデータを送ることもできる。GUIでディレクトリ構造を再現し、ファイルファットエディタでクラスタ番号を変更したり、イメージビューアを使って画像を見ながら復元作業を行える。

データ復元サービスも今後行う予定。



Software　　発売日　2001年1月9日

WindowsからUNIXマシンを操作可能なPC Xサーバ ASTEC-X 3.10

URL　http://www.astec.co.jp/

発売　株式会社アステック・プロダクツ
TEL　03-5804-1853
価格　7万8000円

アステック・プロダクツから、PC XサーバASTEC-Xの新バージョン「ASTEC-X 3.10」が1月9日に発売される。Windows 2000／NT 4.0／Me／98／95上で動作し、ネットワークで接続したUNIXマシン上のX Window System画面をPC上に表示し、操作できる。Windows上の日本語入力システムを使用して、UNIXアプリケーションに入力が可能になる。

今回のバージョンアップでは、SolarisのJava環境に依存している一部のアプリケーションで日本語が表示できない点の改善、CAD／CAMなどのアプリケーションで使われる256色エミュレーションモードでの描画性能を25%向上、Tru64UNIXのCDE環境用フォントの強化などが行われた。





コンパックがLinuxへの取り組みを本格化
2000年12月17日

コンパックコンピュータは、日本オラクルおよびターボリナックスジャパンとのパートナーシップの締結を発表し、Linuxについてこれまで以上に取り組んでいくという姿勢を表明した。

まずはTurbolinuxあるいはRed Hat Linuxをプリインストールした CTOモデル（購入者が仕様を変更できるモデル）を12月20日から販売する。対応モデルは、「ProLiant ML330」（21万6400円～）と「同ML350」（26万7000円～）の2機種だ。

また、2001年の第1四半期に「ソフトウェアアプライアンス」と銘打った、Linuxを採用したアプライアンスサーバを市場に投入する。1Uサーバの「ProLiant DL320」をベースにして、Turbolinuxを採用したWebサーバやMailサーバを内蔵したアプライアンスサーバとなる見込みだ。さらに、同時期に「ProLiant ML370」「同ML530」「同DL360」「同DL380」に、Oracle8iをバンドルしたパッケージを出荷する予定という。これはあくまでバンドルだが、その次の段階として、Oracle8iを内蔵したアプライアンスサーバを検討中ということだ。

米Compaq Computerのソリューション＆ストラテジー担当副社長であるRon Eller氏は、「Compaqは、Linuxをほかのあらゆる OSと同様にサポートしていく。Linuxコミュニティに対して、これまでも積極的に活動してきた。Driverの提供が重要と考えている。オープンソースコミュニティやディストリビューションパートーナーへ提供していく」と同社の姿勢をアピールした。

また、「CompaqはLinuxを搭載したサーバにおいて、26パーセントとシェアが第1位だ。Enterprise分野においても利用が広がり始めている。今後は軽めな利用からミッションクリティカルな分野でも利用が進んでいくだろう」と語った。

コンパックコンピュータは、Linuxを搭載したサーバを販売するほか、Linuxソリューションセンターの本格的な設置やLinux認定技術者を300名体制にするなど、販売体制も強化していくという。

コンパックコンピュータ
(http://www.compaq.co.jp/)

Turbolinuxをバンドルしたゲームソフトを発売
2000年12月9日

ターボリナックスジャパンは、Turbolinux Workstation日本語版6.0（フリー版）をバンドルしたLinux用ゲームソフト5製品を12月8日に発売した（1製品のみ発売日未定）。全国のソフト販売店と、同社のホームページで、15日より出荷を開始した。

今回販売するLinux用ゲームソフトは、米Loki Software社がWindowsからLinuxに移植したもので、都市開発シミュレーションゲーム「SimCity3000（日本語版）」（発売日未定）、3D戦略シミュレーションゲーム「Myth II:Soulblighter」、鉄道会社経営シミュレーションゲーム「Railroad Tycoon 2 Gold」、ファンタジーRPGゲーム「Heroes Of Might＆Magic III」、アクションシューティングゲーム「Quake III Arena」の5タイトル。また、バンドルされるTurbolinuxは、簡易インストールが行え、日本語インストールマニュアルが付属する。価格は、SimCity3000が7900円、Myth II:Soulblighterが4300円、Railroad Tycoon 2 Goldが4900円、Heroes Of Might＆Magic IIIが4600円、Quake III Arenaが7900円。

ターボリナックスジャパン
(http://www.turbolinux.co.jp/)

料理の鉄人とTurbolinux
2000年12月5日

“料理の鉄人”として知られる陳健一氏が技術顧問を務める高級中国料理店「四川飯店」のWebサイトが、Turbolinuxで運営されていることが発表された。

同Webサイトは、“宅配名菜”と銘打って飲茶セット、坦々麺セット、杏仁豆腐、おせちセット、胡麻三昧セットなどを販売するECサイト。

同Eコマースサイトは、Turbolinux Server日本語版6.1をベースに、ワンマーケットが開発したEコマース構築用パッケージ「OneMarket EC for Company Server」を使って構築されたもの。実際のシステム構築は、ターボリナックスジャパンのパートナーであるメディアヴィジョンが担当した。サーバはワンマーケットが契約したデータセンター内に設置されている。

OneMarket EC for Company Serverは、クライアント側のWebブラウザとバックエンドのデータベース間を取り持ち、トランザクションを処理する、いわゆるWebアプリケーションサーバ。商品情報の更新を、Webブラウザベースで簡単に行うことが可能なほか、期間限定のセール価格なども設定できるという。

OneMarket EC for Company Serverの価格は、導入時380万円、保守料2万8000円、ホスティングサービス9万8000円。

四川飯店Webページ（http://ec.i-dk.co.jp/）
ワンマーケット
(http://www.onemarket.co.jp/)
メディアヴィジョン（http://www.mvi.co.jp/）

エー・アイ・ソフト、Linux／FreeBSD版かな漢字変換サーバを無償提供
2000年12月2日

エー・アイ・ソフトは、Linux／FreeBSD版のかな漢字変換サーバ「WXG for Linux／FreeBSD Ver 1.0」の無償ダウンロードサービスを、同社のWebサイト“e言葉”で、11月30日12時に開始したと発表した。

同ソフトは、Cannaの通信プロトコルを採用したかな漢字変換サーバで、MuleやkinpUt2といったCannaのクライアントが利用可能。日本電気のかな漢字変換サーバ「Cannaサーバ」の互換となっており、同サーバに対応したCannaクライアントなら、そのまま利用可能である。ただし、同ソフトは、Cannaサーバと同時には起

動できない。対応OSは、Red Hat Linux、Turbolinux、 LASER5 Linux、 Debian GNU/Linux、FreeBSD、FreeBSD（98）など。

e言葉（http://www.ekotoba.com/）
エー・アイ・ソフト
（http://www.aisoft.co.jp/）

テンアートニ、iDC／xSP向けシステムでインテル、レッドハットと協業

2000年12月2日

テンアートニは11月28日に、インターネットデータセンター市場（iDC）とxSP（ISP／ASP）に向けて、フロントエンドサーバからバックエンドサーバまで統合されたLinuxシステムの開発・提供を推進するため、インテルおよびレッドハットと協業していくと発表した。

3社共同での技術提供、システム開発、マーケティング活動などを通じて、テンアートニでは、初年度10センター、5億円の受注を目指すという。

すでに11月に、インターネット上でサービスを展開する企業ユーザーに対して、サービスの開発から運用までをアウトソーシングサービスとして提供しているイーツのデータセンターにLinuxサーバを納入している。

今回の協業において、テンアートニは、これまでのLinuxシステム構築とWebアプリケーションサーバ開発のノウハウを使用し、約40社のRed Hat認定SIリセラーと共同でiDCのコンサルティング、システム構築、サポートを提供する。インテルは、iDCやxSP向けの市場に対して、ローエンドからハイエンドまでのサーバ製品をインテルアーキテクチャーで実現するための技術情報の提供とマーケティングサポートを行う。レッドハットは、今後最新のディストリビューションにより、iDCで必要とされるクラスタリングなどの高可用性を提供する。

テンアートニ（http://www.10art-ni.co.jp/）

米カルデラ、Linux管理ソフト国際版オープンベータの提供を開始

2000年12月2日

日本SCOは、米カルデラ・システムズ社が現地時間の11月20日に、Linux管理ソフト「Caldera Volution」の日本語を含む国際版のオープンベータの提供を開始したと発表した。カルデラのWebサイトを通じてダウンロードできる。

「Caldera Volution」は、ブラウザおよびディレクトリベースのLinuxシステム用管理製品で、ASPやISP、iDC、ネットワーク管理者など、複数のLinuxサーバやワークステーションを管理する必要のあるユーザー向けの製品。LDAPディレクトリが持つ長所を利用しているという。これを使用することによりネットワーク管理者は、ポリシーやプロフィールを用いて数千台のLinuxシステムを管理できるとしている。また、個々のシステムに直接操作する必要はないという。

「Caldera Volution」国際版ベータのダウンロードサイト
（http://www.calderasystems.com/beta/）
日本SCO（http://www.nihonsco.co.jp/）



サイバーソリューション、ASP向け事業でレッドハットなどと協業

2000年11月30日

サイバーソリューションは11月30日に、丸紅、コンパックコンピュータ、レッドハットの支援を得て、ASP事業者向けに、Red Hat Linuxを中核とするトータルネットワークソリューションの提供を開始したと発表した。これは、e-ビジネス分野に参入を目指すアプリケーション開発業者やベンチャー企業がASP事業を立ち上げるために、さまざまな面から支援するMSP（Management Service Provider）事業に位置づけられる。サイバーソリューションに対し、コンパックおよびレッドハットは製品提供および技術支援を行い、コンパックのハードウェアにRed Hat Linuxを搭載したシステムを丸紅経由で販売する。

ネットビジネスの立ち上げを支援するために、基盤となる設備や機器、システムの動作保証などを提供し、運用開始後は、セキュリティや監視、診断、障害対策などの運用・保守サービスを提供する。データセンターの設備を提供するだけでなく、ASP事業者のアプリケーションと、ハードウェア、Linux、インターネットを組み合わせて実際のシステムを構築し、ネットワーク上にASP事業を展開するための技術支援までをトータルに提供する。

サイバーソリューション
（http://www.cyberspace.ne.jp/）
丸紅（http://www.marubeni.co.jp/）
コンパックコンピュータ
（http://www.compaq.co.jp/）

Wnn7登場

2000年11月29日

オムロンソフトウェアは、京都国際会館で行われた「Linux Conference 2000 Fall」において、日本語かな漢字変換システム「Wnn」の最新版「Wnn7」の概要を発表した。

今回発表されたWnn7では、「Personal Version（仮）」と「Server Version（仮）」の2つの製品を用意し、より一層の使いやすさを追求していくことになる。価格は未定。

Personal Versionの特長は、入力予測、入力補正、連想変換などの入力効率の向上を狙った一連の機能の強化である。また、現在Wnnのフロントエンドとして利用されているxwnmoをGTK+化し、X上のアプリケーションとの連携を強化していく。将来的にはオープンソースとして公開していく意向。

Server Versionでは、複数変換サーバ間の学習情報の同期やIPv6への対応など、より大規模な環境に対応するための機能を実装する。

今後の予定としては、2001年2月～3月にPersonal Versionベータ版をリリースしたあと、2001年5月の製品リリースを目指す。

オムロンソフトウェア
（http://www.omronsoft.co.jp/）

COMDEX 2000 & Linux Business Expo 「超」私的見聞記

～デスクトップLinuxに未来はあるか？

去る2000年11月13日から17日まで、米国ラスベガスで世界最大のコンピュータショー、COMDEX Fallが開催された。COMDEXは昨年からLinux Business Expo and Conferenceと同時開催になっている。20世紀最後となる両イベントのようすを本誌連載「初級Linuxer養成講座」でおなじみの竹田善太郎氏のレポートでお伝えする。

文：竹田善太郎
Text：Zentaro Takeda

Webや電子メールなどによる情報交換が普及したことで、広大な会場で巨費を投じて宣伝・営業活動をするトレードショーは、その存在意義に疑問の声が上がることが多くなった。しかし、「お金の問題は別にして、1年のうちで1回だけ、トレードショーに参加するとしたら？」と問いかけられたら、大多数の業界関係者は「COMDEX Fall」と答えるといわれている。規模、内容の両面において、COMDEXはコンピュータトレードショーのトップに位置していることに変わりはないのだ。そして、その巨大なコンピュータショーに併催されるようになって2年目の「Linux Business Expo and Conference」（以下LBEと略す）も、やはり見逃せないイベントである。

しかし、このような巨大トレードショーの全体像を把握するのは、一個人の力ではとうてい無理である。数十人のスタッフをフル動員しないかぎり、会場全体を見て回ることすら不可能だ。また、そうやってしらみつぶしに取材したところで、どれほどの情報が得られるかは疑問だ。トレードショーの「正しい見方」は人それぞれだろうが、私個人としては、会場の雰囲気を味わいながら、気になった企業のブースで足を止め、時間があれば説明員と会話を楽しむ、というのが合っている気がする。

Microsoftを始めとする「人気企業」のブースは、いつも大勢の来場者でごった返していて、説明員をつかまえるのにもひと苦労させられる。また、このような注目企業の情報は、新聞などの一般マスコミを含めて、ほとんどのパソコン誌やWebサイトなどで詳しく報道されているだろうから、そのようなブースの訪問はあえて避けて、きわめて「個人的な志向」でもってCOMDEXとLBEを「見聞」してきた。

デスクトップLinuxは普及するか？

私が最初にCOMDEXを訪れたのは、今からちょうど10年前のことだが、その当時は「UNIXワークステーション」が注目を集め始め、Sun、DEC、NECなどの大企業が、競ってUNIXワークステーションの製品を発表し、COMDEXの会場でも派手な宣伝活動をしていたのを思い出す。



写真1　Linux Business Expo and Conferenceの会場は、今年からCOMDEXの「サブ会場」である「Sand's Expo and Convention Center」に移され、展示スペースも拡大された。



写真2　LBE展示会場でもっともにぎやかだったブースのひとつが「Red Hat」。最新の「Red Hat Linux 7」を中心に、パートナー企業の関連製品を展示していた。ただし、ギブアウェイ（景品）目当てで行列する「ひやかし客」が多かったのも事実だが……。

そのころは、PCの世界はまだ「Windows 3.1」が登場したばかりで、大多数のPCユーザーはまだDOSアプリケーションを使っていた時代だ。そんな中で、フルカラーのGUIとネットワーク機能を備えたUNIXワークステーションは最先端の存在であり、仕事でコンピュータを使っていた多くのユーザーは、自分の机の上にUNIXワークステーションが置かれる日が来るのを心待ちにしていたはずである。

その後のUNIX業界の変遷については、ここで改めて述べる必要はないだろうが、ユーザー不在の覇権争いの結果として、デスクトップOSの分野はWindowsに席巻されてしまうことになったのだ。

もちろん、UNIXが消滅してしまったわけではなく、UNIXワークステーションなるものは今でも存在するし、サーバ分野でのUNIX OSは健在である。とくに、「e-Business」（この手の「e-なんとか」や「i-なんとか」という言葉は嫌いだが）のサーバ関連では、日本国内はともかく、世界的にはUNIX系OSのシェアはWindowsをはるかにしのいでいる。

しかし、COMDEXの展示会場を眺めてみると、実際には「UNIX」という文字はほとんど目に入らない。その代わり、いやでも目に付いたのが「Linux」の文字である。「Linux」という言葉が、商標として比較的自由に利用できることもあって、Linuxを社名にする企業がものすごく増えて、中にはLinuxとはまったく関係ないのでは、と思われる企業まで現れる始末である（CD-R用のラベルプリンタなどを展示していた）。

そのようなヘンな企業は置いておくとして、現在Linuxがもっとも注目されているのは、各種サーバの分野であることは、もはや動かすことのできない事実になっている。秋葉原で手に入るようなチープなケースに入ったものから、1Uラックに16台ものPentium 4マシンを組み込んだような本格的なものまで、幅広い範囲のLinuxサーバのハードウェアやサーバ関連ソフトウェアが展示されていた。

また、「組み込み」分野での成長も目覚しい。Linux Business Expoの会場には、「Embedded Linux」関連の企業だけを集めた一画が設けられていて、腕時計型PDAからセットトップボックス型の情報端末まで、大小さまざまな製品や開発環境などを見ることができた。

これらに比べて、やはりどうしてもさびしいのがデスクトップ用途におけるLinuxの露出度の低さだ。私も含め、本誌の読者の多くは、デスクトップOSとしてLinuxを使っていて、企業向けのサーバの話よりも、もっと身近なデスクトップ向けの製品に興味があることだろう。しかし、少なくとも「ビジネス」の分野では、まだまだLinuxのシェアは低いといわざるをえない。

それでも、LBEの会場では、Corel、ハンコムなどがデスクトップアプリケーションの展示を行っていたほか、バックアップツール、DVD Videoプレーヤなどのアプリケーション製品が展示されていた。しかし、LinuxをPCにプリインストールした「Linuxワークステーション」のようなものは、COMDEXとLBEの全体を通して、ほとんど見つけられず、目立つのはサーバ製品ばかりであった。また、デスクトップアプリケーションの展示についても、足を止める来場者の数は少なく、心苦しいが「閑古鳥が鳴く」という形容がぴったりのブースも多かった。

DellやIBMなどの大手PCメーカーが、デスクトップPCを含め、自社製品でのLinuxのサポートやLinuxをプリインストールした製品の提供を表明しているのは、すでに報道されているとおりなのだが、中小のPCビルダーにとっては、デスクトップLinux、特に個人や家庭向けの製品は、あまり「うまみのある」商売ではないようだ。結局、個人ユーザーがデスクトップで使うLinuxシステムに関しては、相変わらず、ユーザーが自分でLinuxをインストールして、アプリケーションなどの環境を整備するしかないようだ。

もっとも、このような状況も、2、3年後には変わってくるかもしれない。たとえば、個人向けのデスクトップシステムで需要の高いビデオ関連アプリケーションについては、現時点でLinuxはWindowsに完全に遅れをとっている。しかし、DVDビデオの再生ソフトについては、すでに製品が発表されているし、取材した中には、ビ



写真3　「WordPerfect Office 2000」、「Corel Draw」などのデスクトップアプリケーションを中心に展示していたCorel社。しかし、見学客は少なく、説明員の元気もなかった。Microsoftからの出資受け入れ直後だったのも影響しているのだろうか……。



写真4　Lineo社のスペースで行われていたショーケース形式の「Embedded Linux」コーナー。ホテル向けのテレビ配信システムから、PDAを使ったデータエントリシステムなど、組み込みLinuxの応用製品を数多く展示していた。このほかにも、組み込みLinux関連製品を展示するブースは多かった。

デオ録画や編集などのソフトを開発中というベンダーもあった。もちろん、そのようなベンダーであっても、現実には、製品を出して商売として成り立つかどうかを見極めようとしているところだと思うが、少なくとも技術的な面での障害はなくなっている。あとは、安定性、使いやすさの面でWindows版の同等システムを凌ぐことができれば、すぐにでもデスクトップLinuxの芽は伸びるはずである。

デスクトップLinuxについての2つの見解

COMDEX & LBEでは、参加企業による展示会のほかに、関連企業の経営者や技術者などによる講演や講義も開催されている。特に、「基調講演」（キーノートスピーチ）と題された、企業トップによる講演は、無料で聴講できることもあって、開場前から行列ができるほどの人気企画になっている。筆者もいくつかのキーノートスピーチを聴講したが、その内容については、他のメディアで必要以上に詳細に報道されているので、それを繰り返す必要はないだろう。

ただ、キーノートスピーチは、あくまで「理念」とか「大枠」の話が中心である。具体的な新製品の話が出てきたとしても、それは単なる「宣伝」にすぎない。トップダウン的な思考が得意な人には面白いのかもしれないが、ボトムアップ型の人間にとっては、「大ボラ」か「時代遅れ」のどちらかにしか聞こえない話題が多く、あまり勉強になったためしはない。

それよりも「ためになる」と思うのが、十数種の分野ごとに分けられて開催される「テクニカルセッション」あるいは「カンファレンス」と呼ばれる講義形式のイベントである。話題がより専門化され、それぞれの分野での「現役」の技術者や経営者による講義は、よりいきいきとした内容を聞ける可能性が高い。もっとも、「ハズレ」のセッションも少なくはない。このようなセッションは、展示会の入場料に加えて、かなり高額の聴講料が必要になることも多く、少なくとも身銭を切って来場する人には、お勧めしにくいのも事実だ。

今回は、数多いセッションの中から特に、デスクトップLinuxに関連しそうなセッションをいくつか聴講した。その中で、デスクトップLinuxの将来について、2つの対照的な見解を聞くことができた。

Linuxの普及をさまたげる十の誤解

『Why Not Linux? Ten Misconceptions that Limit Linux's Success』（Linuxを使おう！ Linuxの普及を妨げる十の誤解）と題されたセッションの講師は、UNIX界では知らぬ人はいないであろう人物、John “Maddog” Hall氏（Linux International代表）。このセッションでは、氏の実体験をもとに、おもに企業内でLinuxを導入しようとするときに妨げとなっている、Linuxに対する「いわれなき誤解」を挙げて、そのような誤解に対して反論を展開した。

デスクトップOSとしてのLinuxよりは、サーバOSや開発環境としてのLinuxを中心とした内容だったが、コンピュータやUNIXの専門知識をもたないユーザーの間にありがちな誤解として、「インストールの難しさ」、「信頼性に対する疑問」、「アプリケーションの不足」、「サポートの不在」などの問題に言及したあたりでは、デスクトップOSとしてのLinuxの弱点と、Linuxコミュニティがその弱点をいかに克服してきたかについて述べていた。

たとえば、信頼性の問題については、「所詮、一学生が作ったOS。プロフェッショナルが作ったOSにかなうわけはない」という見方に対して、「Linuxはオープンソースという形態の利点を最大限に生かし、世界中の何百万ものユーザー、何万人ものプログラマーによるテストと改良が行われてきた。企業内や限られた数のユーザーによるテストしか行われていない商用ソフトウェアよりも信頼性は高い」という見解



写真5　GNOMEの開発元となったHelix Code社のブース。ディストリビューションの種類を問わずに、アプリケーションのインストールやアンインストール、アップデートを自動化する「Red Carpet」（2001年1月リリース予定）のデモを行っていた。このような製品は、デスクトップLinuxの分野において、重要な役割を果たすことになりそうだ。

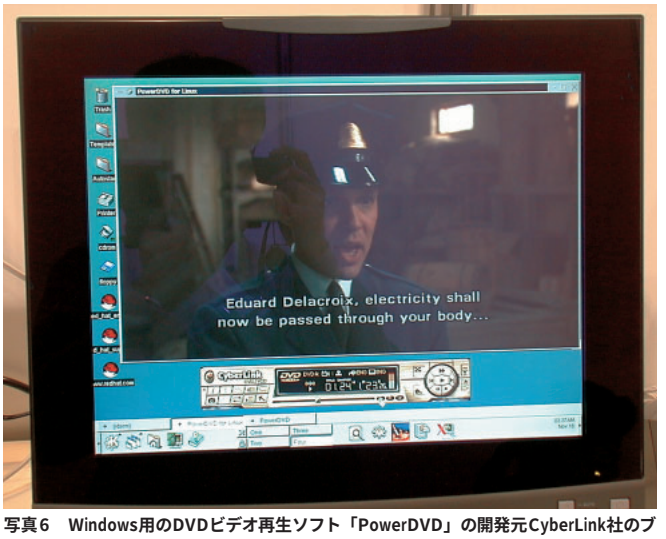


写真6　Windows用のDVDビデオ再生ソフト「PowerDVD」の開発元CyberLink社のブースで、一般客の目に触れない場所で「こっそり」展示されていたLinux版の「PowerDVD」。2001年初頭の製品化を予定しており、完全ソフトウェアによるデコーダ（ハードウェアによるアクセラレーション機能は不要）、Windows版と同等の外観と機能を備えている。写真では、Red Hat Linuxで動いているが、ディストリビューションには依存しないとのこと。同社では、このほかにもLinux用のビデオレコーダソフトやビデオ編集ソフトなどの開発を検討しているということである。

を述べている。また、Linuxはマルチプラットフォームであることに触れて、「単一プラットフォーム（たとえばIntel CPUのみ）のOSよりも、複数のプラットフォームでテストされる過程で、通常では見つからないようなバグが発見される可能性が高く、そのぶん信頼性も高い」と述べた。

セッション全体を通しては、Linuxの普及へ向けた今後の課題を提起するものではなく、これまでのLinuxコミュニティの成果を総括するような内容で、それほど目新しい内容ではなかったが、信頼性や柔軟性について、Linuxが他のOSに引け目を感じることはまったくなく、サーバ、デスクトップなどあらゆる分野で「実用的なOS」として利用できることを結論付けている。

デスクトップOSとしての課題

一方、カナダのバンクーバーに本社を置き、Linux関連の各種サービスの提供を行っているLuteLinux社の社長、Jay Daunheimer氏のセッション『Getting to the Desktop ― Breaking Windows』（デスクトップへの進出－Windowsに打ち勝つ）では、デスクトップ用途において現状のLinuxを冷静に分析し、デスクトップ分野でLinuxがシェアを確保するためには、なにをすべきなのかについて論じた。

2003年の時点におけるデスクトップLinuxのシェアを5%と見積もったうえで、デスクトップOSに求められている要素と、その中でLinuxに不足している部分を列挙した。課題として挙げられた項目は、前出のJohn“Maddog”Hall氏の挙げた「十の誤解」と共通する部分が多く、Linuxコミュニティにおいて、問題意識の「共通化」が図られていることを暗示している。

楽観的なHall氏とは対照的に、Daunheimer氏のセッションでは、デスクトップOSとしてのLinuxには課題が多いことを示唆していた。たとえば、一般ユーザーの間には、「相当に強い動機がないと、新しい環境には移行しない」という傾向があることを指摘し、「WindowsからLinuxへ移行させるには、Windows環境ではできない何かがLinuxにはあることを、一般ユーザーに強く感じさせなければならない」と述べた。また、それに関連して、Linux固有のキラーアプリケーションの追求も重要ではあるが、既存のアプリケーションの「完全な移植」もより重要であると述べている。「安易にエミュレーションに頼るべきではない……。OS自体はWindowsより堅牢で安定しているのだから、同じアプリケーションを移植しただけであっても、より安心して使えるようになり、それはユーザーとって大きな魅力となる」と主張している。

運用の面でも、Linuxのインストールは、以前に比べて格段に簡単になったとはいえ、デスクトップOSとして一般ユーザーを取り込むには「ターンキーシステム」のレベルにならなければいけないことを繰り返し強調していた。また、初心者教育やサポートの面では、Webページや電子メールなどを使って、「技術者」やそれに近い人々が直接ユーザーに対処する方法には限界があり、「教育のプロ」の導入や「電話で直接応対するユーザーサポート」の重要性を指摘した。

このように、多くの問題を指摘しつつも、Linux本体の信頼性や性能については、Hall氏と同様に、オープンソースの利点を強調し、Linuxコミュニティの成果を称えていた。一方で、「自分自身も含め、コアなLinuxユーザーは依然としてコマンドラインを好む。そのようなLinuxコミュニティからは、本当にシンプルで使いやすいデスクトップシステムは生まれないかもしれない」という自嘲も付け加えていた。

両氏のセッションを聞き、そしてCOMDEXやLBEの展示会場の様子を見る限り、現状ではデスクトップOSとしてのLinuxは「まだ芽が出ていない」状態にあると認めざるを得ない。しかし、Linuxコミュニティにおいて、一般ユーザーの立場から見た問題意識の共有が進めば、2、3年後には大きな方向転換もありえるのでは、と感じることができた。





写真7　Agenda Computing社のLinux PDA「Agenda VR3」。その名が示唆するように、NECのMIPS互換RISC CPU VR4181（33MHz）を搭載し、8MバイトのRAMと16MバイトのFLASH ROM上でLinuxカーネルとXを動かしている。デフォルトで搭載されるアプリケーションは、一般的なPDAと同様だが、カーネルや各種ライブラリはすべてオープンソースソフトとして公開されるので、自由にアプリケーションを開発できる。現在「Developer Version」（価格は149USドルから）の販売予約を、同社のWebサイト（http://www.agendacomputing.com/）で受け付けており、2001年1月の出荷開始を予定している。ミニキーボードや通信ケーブルなどのオプションも用意されている（写真右）。日本語版については検討中とのこと。



写真8　John“Maddog”Hall氏のセッションにて。ちなみに、ヒゲとともに彼のトレードマークのひとつだった肩掛けカバンは、今回はデイパックにアップグレードされていた。

Distribution

新着ディストリビューション

LASER5 Linux 6.4

ついにLinuxの年賀状ソフト誕生。LASER5 Linuxは、前バージョンでもWindowsユーザーがオフィス環境で必要とするツールを、MS Office互換のThinkFree Officeでユーザーに提示したが、今回もまたひとつLinuxに欠けていた領域を埋めている。来年の年賀状はLinuxですか？

Omoikane GNU/Linux 1.1

こんにちはDebian。Omoikane GNU/LinuxはDebian GNU/Linuxをベースとするディストリビューションだ。Debianの泣き所であるインストーラを独自に開発し、Red Hat Linuxユーザーも簡単にDebianシステムをインストールできるのが特徴だ。独自パッケージングされたGNOME 1.2も日本語対応が強化されいるので、OmikaneはDebian入門に最適だ。

Plamo Linux 2.1

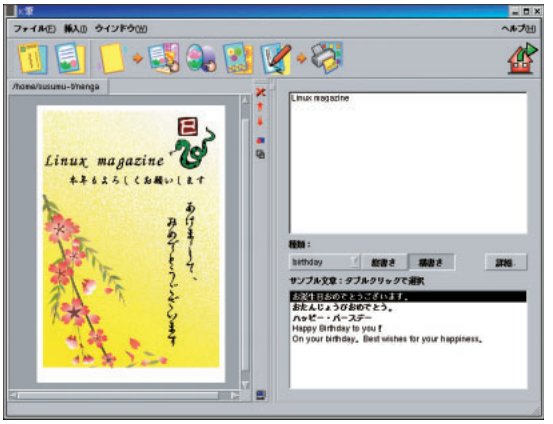
必要最小限というUNIX的思想。アプリケーションはたくさん収録されているけど、どれを使えばいいのかわからない。肥大化が進む昨今のディスビューションを使っているユーザーは、アプリケーションのパッケージでディスクを肥やしているのではないだろうか。Plamo Linuxは、作者のこじまさんが日々の作業に必要だと感じたソフトウェアで構成されている。

LASER5 Linux 6.4

日本の老舗LinuxディストリビューターであるレーザーファイブからLASER5 Linux 6.4（以下、LASER5 6.4）が12月にリリースされた。

LASER5 6.4はRed Hat Linux 6.2をベースにしており、基本システムには、カーネルに2.2.16、XにXFree86 3.3.6、Cライブラリにglibc2.1.3を採用している。また、今回から新しいGNOME 1.2が収録された。

画面1　年賀ハガキ作成ソフトのK筆
ついにLinuxにも年賀状ソフトが登場。というわけで本年もLinux magazineをよろしくお願いします。



製品版のタイプは2種類

LASER5 6.4にはDevelとデラックスの2種類が存在する。

デラックスには、Wnn6やATOKなど日本語デスクトップ環境を充実させる商用アプリケーションが収録されており、Develにはデスクトップの収録物に加えてJBuilder 4 FoundationやCOBOLコンパイラなどの開発環境、マルチブートマネージャSystem Commander Liteが付属する（**表1**）。

Linuxで年賀状を作る

製品版のLASER5 6.4には山田洋行が開発する年賀状作成ソフト「K筆」（**画面1**）が付属する。年賀状作成ソフトといえばこの季節の定番ソフトで、これまではWindowsやMacで動作するものがほとんどであった。このK筆には年賀状用の日本語フォントが含まれているので、Linuxでも凝った年賀状を作成可能だ。

充実の開発環境

LASER5 6.4 develにはGPL配布のTinyCOBOL（http://tiny-cobol.sourceforge.net/）というCOBOLコンパイラが収録されている。COBOLは以前から事務処理プログラムの開発に使われている言語で、その資産を生かそうと、最近再び注目されている。

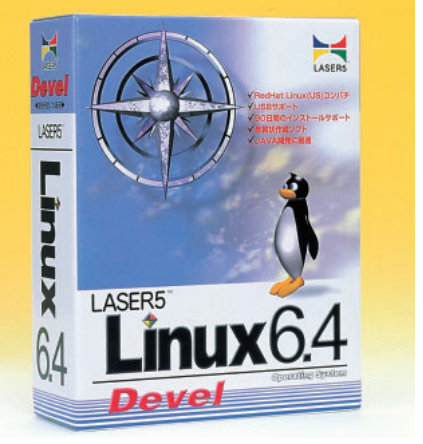
このほかにもGUIインターフェイスを持つJavaの統合開発環境JBuilder 4を収録して、Windowsでおなじみの開発環境を提供している。また、Javaに関しては、Sunが提供しているJDKも収録されている。

これからなにかひとつ言語を学習しようというユーザーに、一通り開発環境がそろっているLASER5 6.4 develはお勧めのディストリビューションである。

本誌付録CD-ROM収録のLASER5 Linux 6.4はFTP版です。非商用ソフトだけが含まれ、製品版を販売しているレーザーファイブからのサポートを受けることはできません。

収録物	概要
Wnn6 Ver.3	日本語入力プログラム
ATOK12 SE for Linux R.2	日本語入力プログラム
K筆	はがき作成ソフト
eWnn	和英辞書引きソフト
翻訳魂	和英・英和翻訳ソフト
ThinkFree Office	MS Office互換をめざすオフィスアプリケーション
開発向けCD	JBuilder 4.0やTinyCOBOLなどの開発環境を収録
Sysrem Commander Lite	マルチブートマネージャの定番ソフト扱えるのはFATとext2のみの機能限定版

表1　LASER6 Linux 6.4のおもな収録物
インストールCDや日本語環境構築に定番の商用ソフトが付属する。DevelにはJBuilder 4.0などを含むCDと、機能限定版のSystem Commanderが収録される。



製品名　LASER5 Linux 6.4
価格　9800円（デラックス）
1万6800円（Devel）
問い合わせ先　レーザーファイブ株式会社
03-5818-6626
http://www.laser5.co.jp/

Omoikane GNU/Linux 1.1

Omoikane GNU/Linux 1.1（以下、Omoikane）がオモイカネ（http://www.omoikane.co.jp/）より12月1日にリリースされた。

OmoikaneはDebian 2.2r1ベースのディストリビューションで、カーネルにバージョン2.2.17、XにXFree86 3.3.6、Cライブラリにglibc2.1.3を採用している。

Omoikaneは現時点で店頭販売されておらず、入手方法はFTPでイメージファイルをダウンロードするか、オモイカネのWebサイト（http://www.omoikane.co.jp/i/ogl-shop/）から実費負担で購入するかのいずれかだ。Webサイトから購入する場合は、CD-ROMのほかに約30ページのインストールガイドが付属する。また、詳細な時期は未定ながら、今後一般ユーザーへのサポート提供も予定されている。

Omoikaneってなあに？

OmoikaneがベースとするDebianで使われているパッケージ管理方式は、一度インストールすれば再インストール不要とまでいわれるほど強力だ。

しかし、Red Hat系のディストリビューションに比べてインストールが難しいためかDebianの普及度はいまひとつの現状である。

そこで、ハードウェアの自動認識に対応した独自開発のインストーラ（**画面1**）を採用して、Debianに不慣れなユーザーも簡単にインストールできるように工夫されている。

Omoikaneは本家のDebianとほぼ完全な互換性を保っているので、約4000のDebian公式パッケージはOmoikaneでもそのまま利用可能だ。一方、Debian 2.2に収録されているやや古めのGNOME 1.0を、オモイカネが独自にパッケージングしたGNOME 1.2と差し替え、フォントサイズを調整したNetscapeを収録して日本語環境を強化している。

パッケージ名	おもな用途
WorkStation	メールの読み書きなど一般的なデスクトップ用
Server	インターネットサーバやファイルサーバなどのサーバ用
Math	OpenXMを収録した計算や論文作成など科学研究用
Beowulf	BLACSを使ったクラスタ環境構築用
X-term	CD-ROMのみでLinuxを使う

表1　Omoikane 1.1の5つのパッケージ
Omoikane 1.1にはユーザーの用途に合わせた5種類のパッケージが容易されている。いずれのパッケージも、バイナリとソースを1枚のCD-ROMに収録しているのが特徴だ。

5種類のラインナップ

Omoikaneには用途によって次のようなインストールCDが用意されている。

■WorkStation

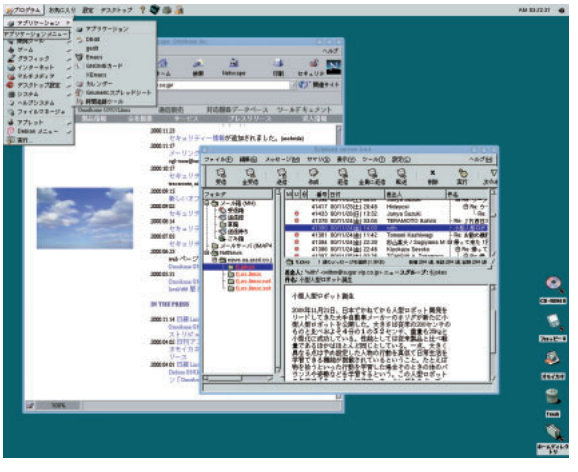
WorkStationはメールの読み書きやWebブラウズなど、デスクトップ用に使うディストリビューションだ。GNOME 1.2を採用したデスクトップ（**画面2**）はMacのようなデザインで、GUIメーラとして定番になりつつあるSylpheedも収録されている。

■Server

Serverはネットワークサーバ構築用だ。qmailやProFTPDといったサーバ用



画面1　Omoikaneのインストール画面
テキストベースだが、ハードウェアの自動認識に対応している。インストールメディアもさまざまだ。



画面2　Omoikane 1.1WSのデスクトップ
Omoikaneでは独自にパッケージングされたGNOME 1.2が標準のデスクトップ環境だ。デスクトップデザインはなぜかMacライクだ。

プログラムのほか、新しいユーザー管理方法としても期待されるOpenLDAPや、動的なWebコンテンツを作成するPHPの最新バージョンPHP4など、最近話題のアプリケーションが意欲的に採用されているのが特徴だ（**表2**）。このほかにも、システム内のファイルが不正に改ざんされたり消去されていないかをチェックするtripwireを収録している。

■Math

Mathは数学研究用のパッケージだ。Mathに収録されるOpenXMは、富士通で開発中のRisa/Asirや、Wolfram Inc.が開発するMathematicaといった数式処理ソフトでの計算を、ネットワーク上のホストで分散処理するためのパッケージである。

■Beowulf

Beowulfは線形代数計算用のクラスタリングパッケージである。これまでの大規模数値計算といえば、Crayなどのスーパーコンピュータがおもなプラットホームだったが、進化の早い最近のプロセッサ事情を反映してか、現在はそこそこのスペックのマシンを多数ネットワークで連動させて計算する手法が盛んになっている。Beowulfに収録されるBLACSは、異なるアーキテクチャのマシンをネットワークで接続して計算を分散処理する際に、各アーキテクチャの差異を吸収するためのパッケージである。

■X-term

X-termはハードディスクが不要のLinuxである。マシンが起動しなくなったときなどに、X-termをCD-ROMから起動してレスキュー用として使うのも有効だろう。

ポリシーを反映したパッケージ

Omoikaneは収録パッケージの選択に独自のポリシーが反映されている。

■TelnetではなくSSH

デスクトップ用のWSにはTelnetのデーモンプログラムが収録されない。Telnetは通信内容をクリアテキストで流すので、クラックの対象となるケースが多い。このため、Omoikaneではセキュリティ面を考慮してTelnetデーモンの代わりに、通信内容を暗号化するSSHが収録されている。

■GRUB

GRUBはポストLILOと目されているブートローダだ。ハードディスクの先頭から約8Gバイトを超える領域にあるOSや、ReiserFSに置かれたLinuxカーネルのブートに対応するほか、プラマリマスタ以外のディスクにインストールされているWindows 9xをブートできるなど、豊富な機能が特徴だ。Omoikaneでは、ベースのDebian 2.2(potato)ではなく、開発版のwoodyに含まれる新しいバージョン（0.5.96.1）のGRUBを収録している。

■自動インストール機能

Omoikaneでは、インストール時にユーザーが入力したネットワーク情報や、パーティションの設定情報などが複数のファイルに保存される。

Omoikaneにフロッピーをセットして、make-autoinstというコマンドを実行すると、Linuxの起動イメージと、初回インストール時に入力した設定情報を含んだディスクが作成される。

このディスクをほかのマシンにセットして起動すると、ほぼ全自動で同じ設定内容のOmoikaneをインストールできるというわけだ。大量のマシンにLinuxを導入する際に有効な機能である。

なお、このコマンドを収録するパッケージautoinst-tools 0.7にはフロッピーディスクを作成できないという不具合があるので、最新バージョン0.8をFTPサイトから入手しよう。

Debian入門に最適

Omoikaneは、Debianとの互換性を保ちつつ、独自のインストーラでRed Hat系ディストリビューションユーザーでも簡単にセットアップ可能だ。また、日本語環境も独自に強化されているので、Debianに興味を持っているユーザーは、OmoikaneでDebianの世界に踏み込んでみてはどうだろうか。

パッケージ名	概要	URL
OpenLDAP	ユーザー管理情報などをネットワーク上で分散管理する	http://www.openldap.org/
PHP4	データベースと連動させて、動的なWebコンテンツを作成する	http://www.php4.org/
qmail	高いセキュリティを保持しつつ高速にメールを配送する	http://www.qmail.org/
Tripwire	システム内にあるファイルの不正な改ざんなどを検出する	http://www.tripwire.org/
OpenSSH	ほかのホストとの通信経路を暗号化する	http://www.openssh.com/
GRUB	8Gバイト超えやReiserFSからのOSブートに対応する	http://www.gnu.org/software/grub/
autoinst-tools	同じ構成のOmoikane 1.1のインストールを支援する	http://www.omoikane.co.jp/
OpenXM	Mathematicaなどの数式処理ソフトで行う計算をネットワーク分散させる	http://www.math.sci.kobe-u.ac.jp/OpenXM/
BLACS	線形代数計算をクラスタ処理する際に、アーキテクチャの差異を吸収する	http://www.netlib.org/blacs/

表2　Omoikaneで採用されているツール

Plamo Linux 2.1

こじまみつひろさんが開発する日本語ディストリビューションPlamo Linux 2.1（Plamo Linux）がリリースされた。

基本システムにはカーネルにバージョン2.2.16、XにXFee86 3.3.5、Cライブラリにglibc2.1.94（glic2.2の開発版）が採用されている。

PC98に対応

Plamo Linuxは、作者のこじまさんがSlackwareを独自に拡張したディストリビューションで、日本語環境構築には古くからLinuxの日本語化に尽力してきたプロジェクトPJEの成果物を利用している。

ベースのSlackwareには、Red Hat LinuxやDebian GNU/Linuxのようなパッケージ管理システムがないので、アプリケーションのインストールはソースをコンパイルして行う。

このためアプリケーションの完全なバージョン管理ができず、ほかのディストリビューションに比べて時代遅れの感があるのは否めない。

しかし、「オレがバージョン管理システムだ」というユーザーにとってはリソースに無駄が出ないので、SlackwareにもPlamo Linuxにも根強いユーザーが存在する。セキュリティホール対策は、なんといっても自分でmakeしてインストールするのがいちばん早い。

また、元国民機のNEC PC98シリーズに対応しているのもPlamo Linuxの人気の秘密だ。

UNIX的環境

Plamo LinuxにはGNOMEやKDEといったデスクトップ環境が収録されておらず、AfterStep（**画面1**）などのウィンドウマネージャでXを使用する。

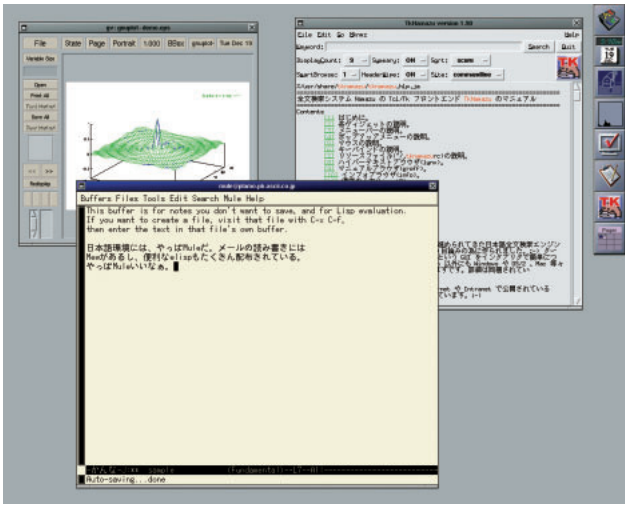
古くからのUNIX系OSユーザーは、オペレーションはコマンド入力、アプリケーションの設定はテキストファイルを直接エディタで編集、というのが一般的な感覚なので、GUI設定ツールが多く収録されるGNOMEなどを必要としないのだ。GNOMEやKDEが収録されていないのは、こじまさんが必要としていないからだろう。

Linuxをいちから勉強する

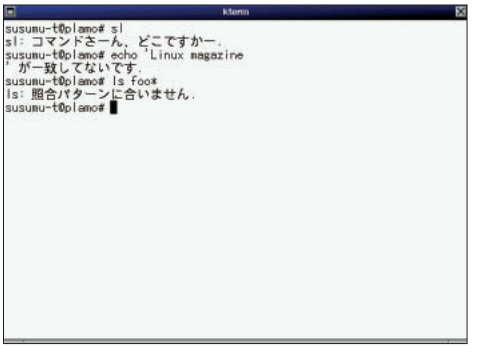
さて、Plamo Linuxはパッケージ管理はできないし、GUIツールもないからLinux初心者には不向きかというとそうでもない。

GUI設定ツールを使えば確かにLinuxに不慣れなユーザーでも簡単に設定できるが、ネットワークやサーバプログラムの設定ファイルがGUIの後ろに隠れてしまうため、必然的にユーザーはLinuxをブラックボックスとして扱うことになる。GUIツールが今後さらに普及しても、書式を理解しながらテキストの設定ファイルを編集することが、なんだかんだ言ってLinuxを使いこなす早道である。

gnuplotで作成したグラフをLaTeXに貼り付けて文書を作成し、Mewでメールの読み書きする、アプリケーションは「make install」でインストールする、Plamo Linuxはそんな昔ながらのユーザーや、Linuxをいちから覚えていこうというユーザーに適したディストリビューションである。



画面1　Plamo Linuxのデスクトップ
Plamo Linuxには3種類のウィンドウマネージャが収録されている（画面はAfterStep）。



画面2　とぼけたシェルのエラーメッセージ
Plamo Linuxではbashではなくtcshが標準のシェルだ。このtcshにはユニークなエラーメッセージが組み込まれている。

Distribution ▶▶▶

並列計算用Kondara MNU/Linux HPC 2000リリース

デジタルファクトリは並列計算用ディストリビューション「Kondara MNU/Linux HPC 2000」をリリースした。Intel版とAlpha版が用意されオープンプライスで2001年1月上旬から発売される。

Kondara MNU/Linux HPC 2000のおもなターゲットは、大規模な数値計算を行う科学・金融分野だ。

このディストリビューションは、プライマリノードマシンにCD-ROMからKondara MNU/Linux HPC 2000をインストールすると、ネットワーク経由で残りのスレーブノードのセカンダリマシンに、プライマリノードのマシンと同じ構成でインストールして、簡単にクラスタリングシステムを構築できるのが特徴だ。

また、プライマリマシンからセカンダリマシン群をシャットダウンできるので、管理面の手間も軽減可能だ。

Kondara MNU/Linux HPC 2000は、カーネル2.2.16、glibc2.1.3、gcc 2.95.2で構成され、並列計算ライブラリにはLAM、MPITCH、PVMが収録されている。

デジタルファクトリ株式会社（http://www.digitalfactory.co.jp/）

Turbolinux Server 6 for zSeries and S/390リリース

ターボリナックスジャパンは、IBMのメインフレームS/390とzSeriesで動作するディストリビューション「Turbolinux Server 6 for zSeries and S/390」をリリースした。価格は15万円からで2001年1月に発売される。サポートを受けるには別途契約が必要となる。

Turbolinux Server 6 for zSeries and S/390は、常時高負荷にさらされるWebサイトがメインターゲットのディストリビューションだ。S/390は1台のマシン上にあたかも仮想的な複数のハードウェアが動作しているような環境を作り出せるので、1台のS/390上にOS/390やLinuxシステムをそれぞれ複数台構築できる。

ユーザーはS/390上のOS/390にIBMのRDBMS DB2を使ってデータベースシステムを構築し、同じくS/390上のLinuxでApacheなどを走らせることができる。稼動率99.999%を誇るS/390にデータベースと連携させたeコマース用のWebサイトを構築する。

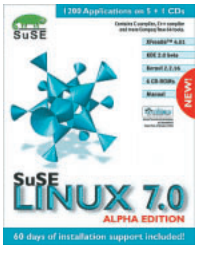
ターボリナックス ジャパン株式会社（http://www.turbolinux.co.jp/）

OpenLinux eServer2.3日本語版の発売元が日本SCOに変更

ドイツのLinuxディストリビューターSuSE Inc.はAlphaアーキテクチャ用の「SuSE Linux 7.0 Alpha Edition」を11月30日にリリースした。

SuSE Linux 7.0 Alpha Editionの販売価格は49.95USドルで、6枚のCD-ROMと約500ページのマニュアルで構成され、60日間のインストールサポートが付属する。

基本システムはカーネル2.2.16、XFree86 4.0.1、glibc2.1.3で構成される。また、Compaqが開発するAlpha用のFortran、C、C++コンパイラと数値計算ライブラリも収録される。



SuSE Inc.（http://www.suse.com/）

Trubolinux OS for Alphaリリース

ターボリナックスジャパンは、Alphaアーキテクチャ用ディストリビューション「Trubolinux OS for Alpha」を12月14日に発表した。

Webアプリケーション開発用キットや、GUI設定ツールが付属し、本体価格は7800円。ユーザーには90日間に限り3件のインストールサポートが提供される。

アクセス数の多いWebサイトやタイトなグラフィック処理を行うサイトがおもなターゲットである。

ターボリナックス ジャパン株式会（http://www.turbolinux.co.jp/）

Red Hat Linux for Itaniumリリース

米Red Hat, Inc.はIntelが次世代プロセッサに予定しているItaniumに向けたRed Hat Linuxのベータ版を12月19日にリリースした。

Red Hat Linux for Itaniumはカーネル2.4を採用しており、同社のFTPサイト（ftp://ftp.redhat.com/pub/redhat/ia64）からダウンロードできる。

Red Hatは開発者などに向けて各種サポートサービスを始めている。たとえば、インストールや設定に関するサポートは、電話対応も含み5インシデントで約1000USドルだ。

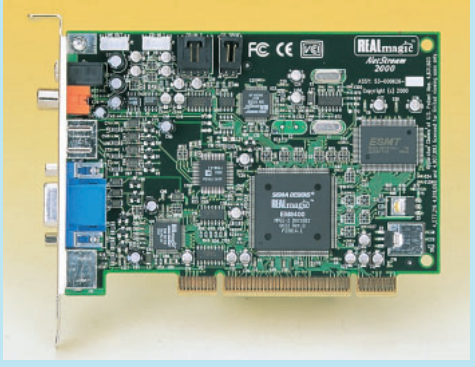
Red Hat, Inc（http://www.redhat.com/）

Products

40 Linuxに対応するハードウェアDVD／MPEG-2デコーダカード
REALmagic NetStream 2000

42 最新プロセッサPentium 4 1.5GHz搭載ハイエンドPC
VC81500UXI Vintage Pro Pentium4プロセッサLinuxモデル

Linuxに対応するハードウェアDVD／MPEG-2デコーダカード



REALmagic NetStream 2000

DVD-ROMドライブの価格もだいぶ値下がりして買いやすくなってきた。DVDソフトも豊富で画質が良いとなれば、Linuxで見たいという要望もあるだろう。個人向けの製品ではないが、Linuxに対応するハードウェアデコーダカードに注目してみた。

製品名	REALmagic NetStream 2000
価格	7万円
問い合わせ先	株式会社バーテックス リンク TEL 03-5259-5126 http://www.vertexlink.co.jp/

バーテックスリンクから、米Sigma DesignsのハードウェアDVD／MPEG-2デコーダカード「REALmagic NetStream 2000」（以下NetStream）が発売されている。

NetStreamには同社の専用LSI「EM8400」（**写真2**）が搭載されており、MPEGのデコードをハードウェアで行う。PCのPCIスロットに挿入し、DVDやMPEG-1／2ファイル、ストリーミングビデオの再生を支援する。

NetStreamはグラフィックスカードからのVGA出力をいったん自身に取り込み、DVD画像を重ねてモニタに送出するアナログオーバレイで表示するようにしている。そのため、グラフィックスカード上のVRAMを書き換えるためのデータ転送をともなうビデオキャプチャカードのような、PCIバスへの負荷もかからない。

また、インターレース方式のDVD映像を、PCモニタのプログレッシブ



写真1　カードのコネクタ部分
左から、オーディオ出力、S/PDIF出力、TV出力、VGA出力、VGA入力。専用コネクタでPCのグラフィックスカードとVGA入力を接続し、VGA出力からモニタへつなぐ。



写真2　MPEGデコーダEM8400
専用LSIでMPEGデコードを行うため、CPUの負荷がとても軽く画質も良い。

スキャンでもきれいに表示する回路も内蔵している。なお、アナログオーバレイという性質上、VGA出力とドットクロックの同期を取るため、最初にキャリブレーションが必要だ。まずオートキャリブレーションしてから、表示エリアを合わせる作業を行う。

Linux用ドライバ

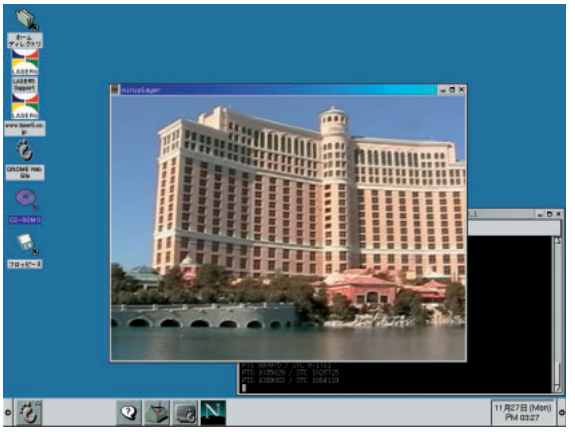
NetStreamには、Windows用のDVDプレーヤソフトが標準で付属しているが、Linuxで利用するためのドライバが、Sigma DesignsのWebサイト（http://www.sigmadesigns.com/support/download_netstream2000_linux.htm）から提供されている。

DVDに記録されているデータは、CSSというスクランブル（暗号化）が行われていて、コピーできないようになっている。Linux用のDVD再生のソフトがフリーソフトとしてリリースされていないのは、著作権法上の問題でCSSを解除するプログラムを公開できないという理由がある。NetStreamのLinux用ドライバは、そのCSS部分以外をGPLでソース公開しているのでライセンス上の問題はない。

Linux用ドライバは**表1**のような複数の単機能プログラムで構成されている。WindowsのDVDプレーヤのようなGUIベースのインターフェイスは備えていない。

プログラム名	機能
loaddriver	デバイスドライバのロード
regioncodechange	リージョンコードの変更
autocalibrate	オートキャリブレーション
miniplayer	VGAのオーバーレイ出力
playfile	DVD、MPEGファイルの再生
remotecontrol	リモコンによる割り込みを受信
celestepapp	OSD（On Screen Display）による再生

表1　Linux用ドライバの機能



画面1　LASER5 LinuxでDVDビデオ再生
miniplayerが表示したウィンドウ上にアナログオーバレイでDVD画像が表示されている。右下のウィンドウでは、playfileを実行してDVDからデータを読み込んでいる。

最初に、loaddriverでデバイスドライバをロードしておき、DVDをDVD-ROMドライブに入れマウントする。次に、miniplayerを起動し、playfileでDVD上のファイルを指定すると再生が始まる（**画面1**）。miniplayer自身は再生機能を持っておらず、単にウィンドウを開いてNetStreamにそのウィンドウの場所を指定するだけである。playfileがDVDからデータを読み込んでNetStreamに渡すことを行っている。DVDを読み込むときにCSSを解除する必要があるが、それは先ほど組み込んだデバイスドライバが担当する。

マルチメディア端末での利用

Windows用のDVDプレーヤのほとんどがソフトウェア再生であり、MPEGのデコードと画面の書き換えをリアルタイムに行うには、かなりのCPU能力を必要とする。しかし、NetStreamを使えば、コマ落ちが起こったりすることもなく、DVD再生中に他のアプリケーションを実行することが可能なのである。

最近、駅やコンビニには、商品の説明をモニタで見ながら注文できたりするマルチメディア（キオスク）端末が置かれている。DVD（MPEG-2）で再生した良い画質の映像を見ることで、商品購入の動機に結びつけたいのだろう。大手コンビニチェーンのローソンのLoppiでは、2000年3月よりLinuxベースのPCに置き換えるが、そこにNetStreamが使用されることになっている。Linuxでの安定した動作と、独自アプリケーションでCSSへの対応が可能なNetStreamは、専用端末での利用に向いている製品といえよう。

MPEGサポートLSI	Sigma EM8400
アナログオーバーレイLSI	Sigma EM9010
ビットレート	最大20Mbps
ビデオ再生	VideoCD、MPEG-2、MPEG-1などのローカルファイル再生。DVDタイトル再生。ストリーミングビデオ再生
オーディオ再生（S/PDIF）	Dolby Digital 5.1ch、DTS、MPEG I／II、PCM
オーディオ再生（Stereo出力）	Dolby ProLogic、MPEG I／II、PCM
バス・インターフェイス	PCIバス（PCI 2.1準拠）
対応解像度	最大1600×1200ピクセル（85Hz）
TV出力	NTSCフルスクリーン。コンポジット、S-Video端子搭載
オーディオ出力	ステレオミニジャック、S/PDIF端子搭載
その他	DirectShow対応、DVD機能、ナビゲーションフル対応

表2　REALmagic Netstream 2000の主な仕様



VC81500UXI Vintage Pro Pentium4プロセッサLinuxモデル

次世代のPentiumプロセッサであるPentium 4が発売された。そのPentium 4 1.5GHz CPUを搭載したハイエンドPCは、従来のPentiumIIIマシンと比べてどのくらいパフォーマンスが高くなったのか、Linuxへの対応も含めて見てみよう。

製品名	VC81500UXI Vintage Pro Pentium4プロセッサLinuxモデル
価格	27万8000円～
問い合わせ先	株式会社日本コンピューティングシステム TEL 03-3821-3200 http://www.jcsn.co.jp/

インテルが11月21日に発表した最新プロセッサPentium 4を搭載したPCが、日本コンピューティングシステム（JCS）から発売された。今回は、Red Hat Linux 7Jをプレインストールした「VC81500UXI Vintage Pro Pentium4プロセッサLinuxモデル」を試用した。なお、試用機はプロトタイプのため、実際に発売される製品とは仕様が異なる場合があることをご了解いただきたい。

Pentium 4 1.5GHzを搭載

Pentium 4は、1995年に発売されたPentium ProからPentiumIIIまでが採用していたP6アーキテクチャに別れを告げ、NetBurstマイクロアーキテクチャと呼ばれる新しい構造を採用している。簡単に特徴を挙げると、新たにSSE2と呼ばれる144の拡張命令を追加し、命令パイプラインの段数を10段から20段に増強。演算ユニットがコア周波数の2倍のクロックで動作し、システムバスクロックを400MHzに向上して最大3.2Gバイト／秒のデータ転送が可能になったといったところである。

現時点でPentium 4に対応するチップセットはIntel 850だけで、メモリにはSDRAMではなく、高価なDirectRDRAMが必須である。

本機に搭載されているマザーボードはIntel製D850GBで、CPUクーラー用ファンの固定部品をマザーボード上からケースに直接ネジ止めするようになっている。そのためのネジ穴が開いているPentium4用ケースを使用している。また、電源は、Pentium4対応のためATX12Vという補助電源コネクタが付いているDELTA製の300Wタイプである。

そのほかの仕様は、**表2**の通りである。JCSでは、すべてのパーツをカスタマイズ可能なので、メモリやハードディスクを増設したり、グラフィックスカードやサウンドカードをいろいろと選ぶことができる。

Linuxの対応は？

ところで、Pentium 4マシンでLinuxは動作するのだろうか。基本的には、Pentium 4の命令セットはx86用と同じなのでそのまま動作するはずだが、Intelの情報によるとカーネル2.4ベースのLinuxで正式サポートされるとなっている。2000年12月現在の時点でカーネル2.4はテスト版であり、ほとんどのディストリビューションで採用しているカーネルは2.2の

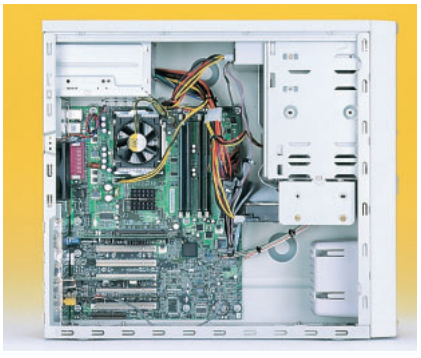


写真1　VC81500UXIの本体内部
少し大きめのファンの下にPentium 4が取り付けられている。電源にはPentium 4対応の300Wを採用。



写真2　Intel製のD850GBマザーボード
チップセットはi850で放熱フィンが取り付けられている。メモリにはDirectRDRAMが必要だ。



写真3　Pentium 4 1.5GHz
高クロック動作が可能な新アーキテクチャを採用。423ピンPGAパッケージで従来のPentiumIIIとの互換性はない。

ため、Pentium 4マシンにはインストールできない場合もあるようだ。

本機にインストールしたのは、Red Hat Linux 7J（カーネル2.2.16-22）だが、Linux起動時のメッセージをよく見ると、「CPU：Intel 00/00 stepping 07」となっていた。そこで、カーネル2.4.0-test12をダウンロードしてきて、make xconfigするとプロセッサタイプに「Pentium-4」が選べるようになっている（**画面1**）。

カーネルを2.4に入れ換えてみると起動時のメッセージに「CPU：Intel（R）Pentium（R）4 CPU 1500MHz stepping 07」と表示された。そこで、/proc/cpuinfoの内容を見てみると**画面2**のように正しく認識されているようだ。

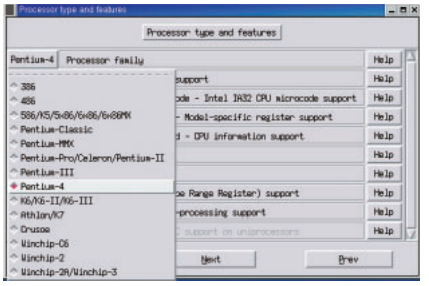
Pentium 4の実力は？

Pentium 4 1.5GHzがどれくらいの実力があるのか、PentiumIII800EBMHzと比較してみよう。ハードディスクやグラフィックスなどは、ハードウェア構成によって異なるので、CPUとメモリ性能を中心に調べてみた。

まず、定番のnbench（http://members.nbci.com/gavrilov_con/nbench/）だ。nbenchはソートやフーリエ変換などを行って、メモリ／整数演算／浮動小数点演算の性能を調べるソフトだ。RPMパッケージからインストールしたプログラムを実行した結果は**表1**のようになった。数値が大きいほうが性能が良いのだが、アプリケーションによっては、ほとんど変わらないものもある。たぶんパイプラインを乱すようなプログラムが不利なのだと思われる。

次に、ベンチマークソフトではないが、SETI@home（http://www.setiathome.ssl.berkeley.edu/）を実行してみた。計算の元になるデータはSETI@homeのサーバから受け取るので計算時間は一定していないため参考程度にしてほしいが、計測時には保存しておいた同じデータを使用して2台のマシンで測定した。結果は、PentiumIIIでは5時間52分58秒だったが、Pentium 4では4時間12分40秒と約72％の時間で終了した。これは実時間で、SETI@homeのWebページに掲載されるCPUタイムはもっと少なくてすんでいる。

Pentium 4をグラフィックスアプリケーションを主とするWindowsで使うのは、現在のところあまりお勧めではないという声も聞くが、Linuxでなら、その拡大したデータ転送能力も生かされるのではないだろうか。



画面1　カーネル2.4.0-test12のxconfig
プロセッサタイプに「Pentium-4」が表示されている。

カーネル2.4を採用したディストリビューションを待ってPentium 4を再度評価してみたい。

```
$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 0
model name      : Intel(R)
Pentium(R) 4 CPU 1500MHz
stepping        : 7
cpu MHz         : 1495.483
cache size      : 256 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 2
wp              : yes
flags           : fpu vme de pse
tsc msr pae mce cx8 apic sep mtrr
pge mca cmov pat pse36 clflush dts
acpi mmx fxsr sse sse2 ss tm
bogomips        : 2981.89
```

画面2　cpuinfoの情報（Pentium 4の場合）
カーネル2.4なら正しく表示される。flagsの最後のほうに「sse2」も追加されている。

テスト内容	Pentium 4 1.5GHz	PentiumIII800EBMHz
数値ソート	465.28	356.88
文字列ソート	40.464	35.372
ビットフィールド	2.3799e+08	1.1339e+08
浮動小数点エミュレーション	48.6	21.132
フーリエ変換	7051.5	7543.9
代入	10.736	5.5446
IDEA	860.21	763.21
ハフマン	585.48	363.43
ニューラルネット	6.0031	5.5424
LU DECOMPOSITION	552.72	329.2
メモリインデックス	6.323	3.789
整数インデックス	4.546	2.976
浮動小数点インデックス	7.229	6.057

表1　nbenchの結果を比較

CPU	Pentium4 1.5GHz
マザーボード（チップセット）	Intel D850GB（i850）
RAM	128MバイトDRDRAM（PC800、最大2Gバイト）
ハードディスク	46.1GバイトIDE（UltraATA/100、7200rpm）
CD-ROM/DVD-ROM	16倍速DVD-ROM（40倍速CD-ROM相当）
フロッピードライブ	3.5インチ1.44Mバイト
グラフィックス	nVIDIA RIVA TNT2 M64 AGP 32Mバイト
ネットワーク	10/100BASE-T×1ポート（21143）
サウンド	Sound Blaster Live! Value
インターフェイス	RS-232C×1、パラレル×1、PS/2×2ポート
PCIスロット	5スロット
キーボード／マウス	日本語109キーボード、3ボタンマウス（PS/2）
サイズ（mm）、重量	200（W）×477（D）×420（H）、約12.8kg
電源	300W（Pentium4対応）

表2　VC81500UXI Vintage Pro Pentium4プロセッサLinuxモデルの主な仕様

箱の中のペンギンたち



文：みわよしこ
Text : Yoshiko Miwa

第2回
組み込みLinux製品の現在

コンピュータを「組み込む」とは何をどうすることなのかは前回説明しましたが、もう一度、簡単に説明しておきましょう。

機器に組み込むコンピュータには多彩な機能は必要なく、ごく限定された機能が求められます。たとえば炊飯器であれば温度や加熱時間の制御、自動販売機であれば商品の温度や個数の把握、投入された金額の判断、お釣りの計算、売り上げの把握ができれば十分なのです。

機器にコンピュータを「組み込む」とは、その用途に必要かつ十分と考えられるレベルに限定し、使用条件（気温・水滴や水蒸気のある場所かどうか……）に適した実装を行うことです。

多彩な組み込み製品を産み出すLinuxのポテンシャル

Linuxには、ほかのOSと比べて以下のような特徴があります。

・機能が限定しやすい

Linuxカーネルはもともと非常に小さいものです。また、機能を限定する場合には必要最小限のモジュールだけでカーネルを構築すればよく、必要な場合のみ動的にモジュールをリンクすることが可能です。このようなLinuxの特徴は、機器への組み込みで機能を限定するのに有利です。

・CPUの選択肢が広い

Linuxを動作させられるCPUは多種類にわたります。パソコンで一般的なIntel CPU（x86）、しかも80386からPentiumまではもちろん、他社製の互換CPUも同様に利用できます。さらに、Power PC、Alpha、MIPS、StrongARMなど、多種多様なアーキテクチャに基づくCPUでLinuxは動作します。

・多様な入出力への対応

Linuxは、シリアル、パラレル、SCSI、USBなど、一般的なバスのほとんどに対応しています。多様な入出力への対応がこれによって可能となります。また、Linuxカーネルは最初からネットワークに対応しているので、ネットワークを利用した通信や入出力も容易に行えます。

・オープンソース

商用の組み込みシステム向けOSと異なり、Linuxにはライセンス料がかかりません。またカーネルを含めてソースコードが公開されており、カーネル以外の改変・再配布も可能です。プログラマーは少ない制約のもと、自由な発想でソフトウェアやシステムの開発を行うことができます。

さらに組み込み用途では、



- **リアルタイム性**
- **動作途中に電源をオフできること**

が求められます。組み込み向けのLinuxディストリビューションは、何らかの形でこれらの条件をクリアしています。

Linuxは開発者やユーザーの求めに応じて、どのようにでも形を変える変幻自在のOSです。このことは、「私をこのように使ってください」という方向性をユーザーに示しているWindowsなどと比較するとさらにわかりやすいでしょう。こうしたLinuxの特性を利用したユニークな製品が、すでに数多くのベンダーから発表・発売されています。

それでは、2000年12月現在までに発表されている製品から、筆者が「ピン」ときた製品をひとつずつ紹介いたしましょう。

IBM LinuxWatch

2000年8月、「腕時計サイズでX Window Systemまで動く！」ということで話題を呼んだ製品です（**写真1**）。あくまで試作品で市販の予定はないようです。

サイズは56mm（幅）×48mm（長さ）×12.25mm（厚さ）と、腕時計としてはちょっと大きすぎる感じです。1970年代、LEDで時刻を表示するデジタル時計が市販され始めたばかりのころ、やはり厚さが1cm以上あったのを思い出します。

画面は白黒液晶のタッチパネルとなっています。さらに画面右側にはローラー・ホイールがついており、操作はこれらを利用して行うことになります。

IrDAとRFの2種類のワイヤレス通信インターフェイスを備えているので、パソコンや携帯電話を始めとした、ワイヤレス対応機器との通信が可能です。電子メールのメッセージを表示できるほかに、ポケベルメッセージを直接受信することができます。カレンダー、アドレス帳、ToDoリストなど、PDAに最低限必要なアプリケーションも備えています。

画面の大きさや操作手段から、スケジュールやアドレスの整理はパソコンで行ない、パソコンから必要なデータだけを配信して持ち歩くという利用方法に限定されますが、この「腕時計」の延長には、たとえば初期のウルトラマンシリーズに登場した「本部と通信できる無線テレビ電話が付いたPDA」といったものもありそうです。

ちなみに搭載しているメモリは、8Mバイトフラッシュメモリと8MバイトDRAMメモリです。Windows CEをOSとして採用しているカシオのCASSIOPEIA E-503が、32MバイトのROMと32MバイトのRAMを搭載しているのと比較すると対照的です。

`http://www.jp.ibm.com/NewsDB.nsf/2000/08081`

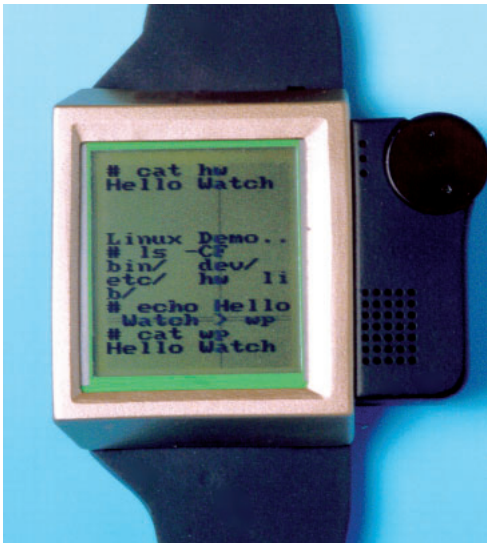


写真1　IBM LinuxWatch
腕時計サイズ（というか、腕時計そのもの）でX Window Systemまで動く。

Kerbango 100E Internet Radio

インターネットでストリーム配信される「ラジオ番組」、つまりRealAudioデータを再生することができるラジオです（**写真2**）。10BASE-Tのインターフェイスが付いており、LANに接続して利用します。

この製品も、OSにはLinux（モンタビスタのHard Hat Linux）を利用しています。搭載しているメモリは、16MバイトSDRAM、8Mバイトフラッシュメモリ、4MバイトROM



写真2　Kerbango 100E Internet Radio
ちょっとコンピュータとは思えないデザイン。インターネットラジオだけでなく、地上波も聞くことができる。

です。ネットワーク配信されるデータをリアルタイムに再生するだけなら、たったこれだけのメモリ資源で可能なのです。

ちなみにこのInternet Radioは、AM/FMチューナとアンテナが組み込まれていて、ごく普通の地上波のAM/FM放送の受信もできます。開発者の「遊び心」を感じます。価格は2000年12月現在、「amazon.com」で299.99USドルとなっています。

このInternet Radioに、MP3の音楽データをダウンロードして再生する機能を追加し、いわば「インターネットラジカセ」と呼ぶべき製品にするのはそう難しくないでしょう。近い将来にそのような製品が開発・発売されることに期待したいところです。

http://www.kerbango.com/

ノキア Mediascreen

製品の特徴をひと言で表すと、「テレビとインターネットと携帯電話のハイブリッド」となります。12インチの液晶パネルにアンテナが付いた、シンプルな外観です（**写真3**）。一見した印象は、大きめの液晶テレビです。現在はまだ試作品の段階です。日本では携帯電話のメーカーとして有名なノキアですが、このような製品も開発しています。

このMediascreenでは、地上波のデジタルテレビ放送（DVB-T）と、Webで提供されるコンテンツを見ることができます。インターネットに接続するには、Bluetoothを利用してLANに接続するか、もしくは組み込んだ携帯電話を利用することになります。

ノキアは車中やキャンプ場など、レジャーの場での利用を想定しているようですが、筆者は自宅用に1台欲しいと思ってしまいました。テレビ・Webコンテンツを「見る」用途をひとつにまとめたこのような製品は、まさに日本人の住居事情のために開発されたようなものではないでしょうか。

ただし日本のBSデジタル放送は、日本独自規格のISDB-Sです。このMediascreenが製品化されたあと、さらに日本向け製品が開発・発売されるまでには、海外のすぐれた製品を日本で発売する場合にしばしば持ちあがる、「独自規格の壁」が問題になりそうです。

http://www.nokia.com/multimedia/mediascreen.html

Agenda VR3

スケルトンの外観が魅力的なパームトップサイズのPDAで、OSにLinuxを採用しています（**写真4**）。メモリ構成は8MバイトRAMと2Mバイトフラッシュメモリ（4Mバイト／8Mバイトのモデルもあり）となっています。

外部との通信手段としてIrDA、シリアルの2つを備えています。外付けキーボードなどのアクセサリも発売されています。価格は最も安価な2Mバイトフラッシュメモリのモデルで149USドルと、購入しやすく抑えられています。アドレス、ToDo、メモ帳といった機能だけでなく、メーラ、FAX対応（オプションのモデムが必要）、ターミナルが最初から含まれているあたりに、LinuxをOSに採用したからこその特徴が感じられます。

スペック的に近いPalm Computing m100（日本国内で



写真3　ノキア Mediascreen
モバイルLinuxのようなイメージだが、テレビも見られる。ソニーのエアボードに似たコンセプトの製品。



写真4　Agenda VR3
一見するとPalmのような外観だ。上部に光るペンギンロゴがLinuxをOSに採用している証だ。





の希望小売価格は1万9800円）と比較してみるのも面白いでしょう。

なお、本製品はまだ日本語化されていませんが、Linuxベースですので日本語化にはそれほど困難はないと思われます。同製品のサイトでは、いくつかのアプリケーションが公開されてダウンロード可能になっていますし、ユーザーがそこにアプリケーションをアップロードすることも可能です。日本語化のためのパッチや日本語化されたアプリケーションがそこに置かれるのも、そう遠い先のことではないかもしれません。

http://www.agendacomputing.com/

ミュージックポッド

ミュージックポッドは、ネットワークを利用した音楽の自動販売機です（**写真5**）。CDショップ、書店、アミューズメントパークなどに設置されているミュージックポッドにMDを挿入し、代金を入れ、購入したい曲を選択すると、それがMDに書き込まれるというシステムになっています。自動販売機への商品の補充に相当する部分が、ミュージックポッドでは光ファイバを利用したネットワーク通信に置き換えられています。曲のダウンロードに関して、一般ユーザーがインターネットからダウンロードするのであれば4分程度かかるところが、30秒程度で済むようにサーバネットワーク構成に十分な配慮がされています。

このシステムを利用すると、CDの製造や流通に関わるコストが大きく削減されるため、音楽愛好者はCDを購入するよりも安価に曲を購入することができます。現在、1曲あたりの価格は150円～300円となっています。歌詞カードは別料金となっており、購入するかどうかを選択できます。このような柔軟さは、CDなどの媒体による流通には望みがたいところです。

ミュージックポッドは、販売店にとっても数多いメリットのあるシステムです。通常のCDであれば、店の規模によって置ける枚数が限られます。小規模店舗では少ない枚数しか置けません。そして、在庫管理も大変な仕事のひとつです。これらの問題は、ミュージックポッドの導入で消滅してしまいます。

また、曲を提供する側の人々にももちろんメリットがあります。CDの制作・流通に関するコストが不要になります。また廃盤となった音楽データも流通させることができます。これは古い曲・マイナーな曲を愛好する音楽愛好者にとっても有益なことでしょう。

実はこのミュージックポッドは、おそらく現在、日本で最も一般の人々に親しまれている組み込みLinux製品です。採用されているLinuxディストリビューションは、ゼータビッツの生越昌己氏が開発した「Zeta Linux」を使っています。ミュージックポッドは現在、日本を代表する組み込みLinux製品と言っても過言ではないかもしれません。

今回のインタビューでは、生越氏にZeta Linuxの開発ストーリーを語っていただきました。

ミュージックポッドによる音楽配信サービスを行っているブイシンクのサイト

http://www.v-sync.co.jp/

ミュージックポッドの配置場所や音楽データに関する情報のあるサイト

http://www.musicpod.co.jp/



写真5　街頭にあるミュージックポッド
多くの人はLinuxが使われているのを知らない。



画面1　ミュージックポッドのWebサイト（http://www.musicpod.co.jp/）。ランキングや、使い方などの情報を掲載している。

ゼータビッツ取締役開発部長・日本Linux協会会長 生越昌己氏に聞く

Interview



生越昌己氏は、日本Linux協会会長を務めています。1990年代初頭という時期にLinuxの価値を認め、日本で普及させたことに関する生越氏の功績が忘れられることはないでしょう。

生越氏はもともと、長く豊富な経験を持つシステムエンジニア・プログラマーでもあります。今回は、組み込み向けディストリビューション「Zeta Linux」の開発ストーリーを通じて、エンジニアとしての生越氏にインタビューを行いました。

――こんにちは。今日は、組み込みシステムをターゲットのひとつとして、生越さんが開発されたZeta Linuxのお話を伺いたいのですが……。

生越：組み込みシステムに誰が興味持つんですか？

――は？

生越：だって一般の人が組み込みをやるようなことはあり得ないでしょう？　液晶は今、入手がとても難しくて、メーカーが10万枚単位で予約して半年後にやっと入手できるような状態だし。

――そういった話はエンジニア以外の人にとっては非常に新鮮だと思います。Linuxディストリビューションの開発が、具体的にどういう作業をすることなのかを知っている人もあまりいませんし。

生越：それはそうですけど。

――まず、開発スタイル※1からお話しいただけますか？

生越：1人でLibrettoをターゲットにしこしこやりました。

――工数※2はどのくらいかかりました？

生越：1人／月くらいです。とりあえず動いて、konとEmacsとtelnetが動かせるようになるまで2週間くらい、ミュージックポッドに載せて使えるようになるまでに1カ月かかりました。



写真　ゼータビッツ取締役開発部長　生越雅巳氏（おごしまさみ）
1963年生まれ。松江工業高等専門学校電気工学科卒業後、メーカ系ソフトハウス、放送局に勤務し、プログラミング、システムエンジニアリング業務に従事。1997年5月よりネットワーク応用通信研究所（現在はゼータビッツ）に勤務、現在は取締役開発部長。また、1992年よりLinuxや386BSDを個人的に使用し、日本国内での普及に尽力した。日本Linux協会では1999年4月の設立以来、会長を務めている。

――意外に少ない工数でできたんですね。

生越：取りかかる前に若干の試行錯誤をやっていたので、取りかかってからの工数が減らせたということはありました。それに小さいですし。

――期間はどのくらいかかったのでしょうか。

生越：10カ月程度です。これに専念していたわけではありませんから。

――開発のきっかけを教えてください。

生越：取り掛かったのは1999年5月です。会社の業務のつもりで始めました※3。もともとWindows CEマシンにLinuxを載せようと思って、カーネル以外のユーザー環境を作っていたのですが、その延長で組み込み用途にも使えるものを開発したんです。Windows CEマシン向けのカーネルの開発プロジェクトは別にありますし。

――ユーザー環境を開発するのだから、PC/AT互換機のLibrettoで開発を行っても別に構わないということですね。Windows CEマシンに搭載する時は、Windows CEマシン用のカーネルに開発したユーザー環境を追加すればよいと。

生越：そういうことです。

――現在はどういう環境で利用可能になっているのでしょうか？

生越：Windows CEマシン（MIPS、StrongARM、SH3）で利用できるはずです。といっても、開発したのはユーザー環境なので、プロセッサは意識しなくてよいはずなのですが。

――作業はどんなふうに進めて行ったんですか？

生越：詳しくはWebサイトに『Zeta Linux 進捗日記（http://www.nurs.or.jp/%7Eogochan/ZetaLinux/diary.html）』があるのでそちらを見ていただくとして（笑）、まず東芝のLibretto 50にメモリ16Mバイト、ディスク64Mバイトの厳しい環境を作りました。その環境にインストール可能な既存のLinuxディストリビューションにはPlamo Linuxが



あったので、Plamo Linuxをベースにすることにしました。プログラミングに際しては、Debian GNU/Linuxのソースコードも参考にしています。ただし、今はPlamo LinuxともDebian GNU/Linuxとも全く無関係になっていますが。

――現在、どういう機能が実現されているのでしょう？

生越：展開して9Mバイトという小さいサイズですがXが動きますし、その上で利用できるターミナルソフトも含まれています。ネットワーク、無線LAN、PCMCIAをサポートしているので、ネットワーク利用には十分な機能を持っていますし、telnet、ftpも利用できます。konも入れていますので、Xでもコンソールでも使えます。

――展開して9Mバイトというサイズは、どういうことから決めたのでしょうか？

生越：コンパクトフラッシュでの利用を前提にしました。

――『Zeta Linux進捗日記』を見ますと、1999年5月当時、Plamo Linuxのライブラリがlibc5だったということでだいぶ苦労されたようですね。

生越：ええ。とりあえずlibc5、libc6の混在した環境を作って動かして、そこからlibc5を追い出して、ひとつずつツールを追加しては動くかどうかを検証していきました。でも、イマイチうまく行かないんで、またlibc6でゼロから作り始めて……。

――libc6にこだわった理由は何でしょうか？

生越：世の中的にはlibc6が標準ですからね。組み込みの専門家は、サイズが大きいlibc6よりlibc5を愛好する傾向があるんですが、組み込みシステムだけではなく、大型汎用機のようなマシンでも利用できるものが開発の目標でしたからlibc6を採用しました。

――ライブラリが足りなくて動作しない場合にはどう対処しましたか？

生越：ツールが現在入れていないライブラリを要求してきたら、そのツールの動作に本当にそのライブラリが必要なのかどうかを検証しました。

――本当は必要のないライブラリを要求しているケースは時々ありますね。ライブラリの問題で入れられなくなったツールはありますか？

生越：ktermはAthena Widget（libXt）を要求するので外しました。代わりにターミナルソフトとして、libcだけで動くrxvtを入れています。

――組み込み用途だと、「いつでも電源のオン／オフが可能」という要請がありますけれども、その部分はどうやって開発したのでしょうか？



画面　ブイシンクのWebサイト（http://www.v-sync.co.jp/）。ミュージックポッドについて詳しい情報が掲載されている。

生越：そこはあまり意識していません。別の仕事で、スワップ領域を利用したサスペンド／レジュームを開発したので、それと組み合わせればよいだろうと考えていました。

――あくまで、Zeta Linuxで開発したのはユーザー環境ということですね。

生越：そういうことです。

――Zeta Linuxを搭載した製品には何があるでしょうか？

生越：ミュージックポッドがあります。Linuxを搭載した組み込みシステムとしての出荷実績を作ったのは、国内ではミュージックポッドが最初だと思います。

――今後、どのような展開を予定されていますか？

生越：基本部分はできあがっていて、組み込みには十分なのですが、アプリケーションを開発する必要があります。Windows CEマシンにインストールして一般ユーザーが利用する場合には、Microsoft ExcelやMicrosoft WordのWindows CE版に相当するアプリケーションも必要になりますし。もっともLinux環境にはすでにgnumericがあり、Microsoft ExcelのWindows CE版よりはずっとMicrosoft Excelに近い機能を備えていますから、それほど困難もないでしょう。

――今日は開発の実作業を主に、興味深いお話をいろいろ聞かせていただきありがとうございました。

※1　人員（たとえば1人なのかチームなのか）、体制（きっちりした組織を作るのか、Linuxのように緩やかな協力体制を作るのか）など、開発参加者のワークスタイルを広く指す言葉。
※2　システムやソフトウェアの開発に必要な人的資源を、人数×日数＝人／日（または人／月、人／年）という単位で表わしたもの。
※3　現在はゼータビッツの製品であることから『Zeta Linux』という名称となっている。社名の「ゼータ」は「10^{21}」という意味の接頭語zettaから取ったものだが、「Zガンダム」にもちなんでいるとのこと。

Linux活用テクニック

「小技・ウラ技」

Linuxを使いこなすためのHint&Tipsを伝授しよう。

文：編集部
Text : Linux magazine

かくし技」

- 知っておきたい、デスクトップ環境を活かすちょっといい技　52
- おさえておきたい中級技（X Window System編）　66
- Linuxのあんな技こんな技（コマンド編）　70
- bash、vi、Emacsの基本技　78

知っておきたいデスクトップ環境を活かすちょっといい技

GNOMEやKDEの急激な進化によって、クライアントPCでのLinux利用も身近になってきた。KDEもGNOMEも、市販のOSと肩を並べるほどの利便性や美しい外観を持ち合わせているうえに、非常に細かなカスタマイズが行えるのが特徴だ。このセクションでは、少し手を加えることで、便利になったり、かっこよくなったり、なんとなく楽しくなったりする技を紹介しよう。カスタマイズに万人向けのセオリーはないが、「こんなことやっている奴もいる」例として眺めていただきたい。

デスクトップカスタマイズ（GNOME編）

GNOMEは、各アプリケーションに共通のLook & Feelを提供し、扱いやすいデスクトップ環境を提供するシステムだ。最新のバージョン1.2は、とても安定しており、使い勝手も優れている。

GNOMEは、見た目や操作性をユーザーの好みに合わせてカスタマイズすることができるが、意外とデフォルトのままで使っているというユーザーが多いようだ。

ユーザーインターフェイスは、ユーザー自身が使いやすいと思えればいいのであって、万人に最適な設定というものがあるわけではない。とりあえずいろいろと試してみて、自分に適した設定を見つけよう。

■ポイントは、コントロールセンター

GNOMEではテキストの設定ファイルを直接扱うことなく、各種の設定が可能になっている。ポイントは、画面左下の足跡マークのメニュー内や画面下部のパネルにある「GNOMEコントロールセンター」だ。コントロールセンターは、左側にカテゴリーごとに分類された項目があり、それぞれの項目を選ぶことで、右側に設定画面が表示されるようになっている。なお、このカテゴリーの分類方法は、各ディストリビューションによって多少異なっている。以下、Vine Linux 2.1を元に説明を行っている。

■セッション管理

X Window Systemでは、セッションマネージャと呼ばれる、一連の作業（セッション）を管理するプログラムが動作している。GNOMEを用いている場合は、ログインからログアウトまでが1つのセッションであり、gnome-sessionと呼ばれるプログラムがセッションマネージャだ。gnome-sessionは、ログアウト時にデスクトップの状態を記憶することが可能で、次にログインしたときにその状態を再現してくれる。ログアウト時に表示されるダイアログに、「現在の設定を保存」という項目があるが、これをチェックしておけばいい。

ログアウト時に、いちいちチェックを入れずに済ませるには、「セッション」カテゴリーから「スタートアップ」を選択し、「セッションの変更を自動的に保存する」をチェックすればいい。

終了時のダイアログから、「現在の設定を保存」というチェックボックスがなくなっているはずだ。さらに、「ログアウト時のプロンプト」をオフにすると、ダイアログも表示されずにディスプレイマネージャに戻るようになる。

■パネル上のアイコン

GNOMEでは、パネル上にプログラム起動用アイコンや、アプレットと呼ばれるプログラムを好きなように置いて、カスタマイズできる。数が増えてきたら、きちんと整理しよう。

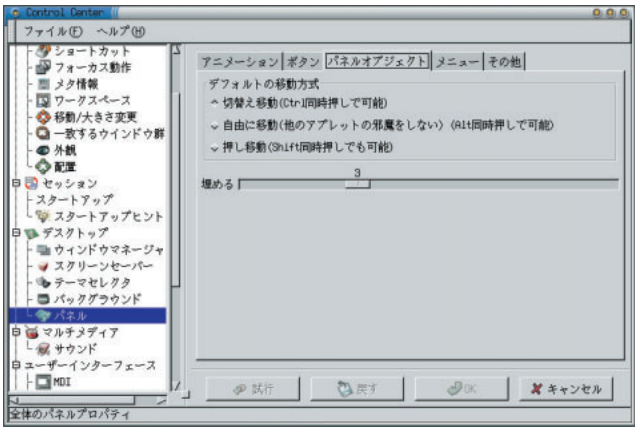
並べたアイコンやアプレットを移動させるには、中ボタンでドラッグすればいい。デフォルトでは、移動させたアイコンが隣のアイコンとぶつかると、位置が入れ替わるようになっている。並べ変えたいときにはこれでも便利なのだが、ほかの動かし方も用意されている。

「デスクトップ」カテゴリーのパネルを選び、「パネルオブジェクト」というタブを選択すると、動かし方が全部で3通り用意されていることが分かる（**画面1**）。デフォルトは「切替え移動」だ。「自由に移動」にすると、ほかのオブジェクトに影響を与えることがなくなる。「押し移動」は、複数のアイコンをまとめて動かすことができる。見た目に一番楽しい方法だ。どの方法を選んだ場合でも、“ Ctrl ”、“ Alt ”、“ Shift ” キーのどれかを押しながら動かすことで、一時的に別の方法で動かすこともできるので、状況によって使い分けるといいだろう。

パネル上にアイコンを追加するには、「パネルに追加」－「ランチャ」を選択し、設定していけばいいが、メニューから選んだ項目をパネル上にドラッグ＆ドロップするのも、よりお手軽でお勧めだ（**画面2**）。

■ツールチップス

これはGNOMEの機能というより、ウィンドウマネージャであるSawfishの機能だが、Sawfish＋GNOMEの環境では、



画面1　パネル上のオブジェクトの移動方法を指定する
「押し移動」を選択すると、複数のオブジェクトを同時に移動させられる。

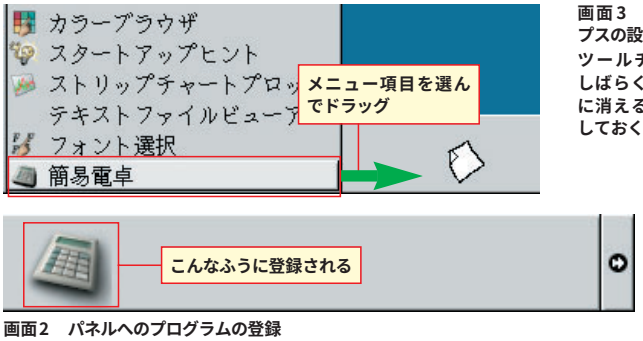
タイトルバーやウィンドウのフレーム（枠）にマウスカーソルを合わせると、ツールチップスと呼ばれるヘルプが表示される。ツールチップスは、わりと大きいため、かえってウィンドウ内部が見えなくなって、「本末転倒だろう！」と突っ込みを入れたくなることもある。

そんなときは、「Sawmillウインドウマネージャ」カテゴリーの「その他」（**画面3**）で、「Tooltips」タブ内にある、「一定時間後にツールチップを消す」にチェックを入れておけばいい。しばらくしたら消えてくれるこの奥ゆかしさがいい。

■コントロールセンターもカスタマイズできる！

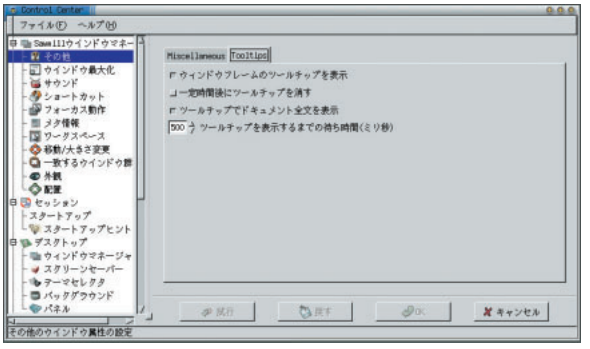
GUIで設定できるのは、お手軽だ。でも一応、どのような仕組みでコントロールセンターが動いているのか知っておくと、より豊かな人生を送れるだろう。たぶん……。

コントロールセンターの左側に表示されるカテゴリーは、/usr/share/control-centerディレクトリ内の構造が、そのまま反映されている。その中にあるファイルには、複数のロケールデータ、実際に起動するプログラム、アイコンなどが記述されている。



画面2　パネルへのプログラムの登録

画面3　ツールチップスの設定
ツールチップスは、しばらく表示した後に消えるように設定しておくのが快適だ。



ミッドナイトコマンダーをおさえておく

ミッドナイトコマンダーといってもピンとこない方もいるかもしれない。GNOMEのメニューにあるファイルマネージャのことである。正確には「GNOME Midnight Commander」、GMCと呼ばれるGUIユーティリティだ。GMCの元になったCUIバージョンの「ミッドナイトコマンダー」もある（**画面4**）。

GMCはデスクトップに配置されているアイコンの表示と操作も制御しているエラいプログラムで、GNOMEの起動時には必ず立ち上がるようになっている。「GMCを制する者はGNOMEデスクトップを制す」と言っても過言ではないだろう。それは大げさにしても、GMCの操作を覚えておけば、たいていのファイル操作は行えるし、うろ覚えのコマンドオプションで迷うこともなくなる。これを機会にぜひ押さえておいてほしい。

GMCの初期画面はGNOMEを使ったことがある方なら、一度は目にしたことがあるはず。メニューバーに並んでいる各メニューとメニュー内の各項目は、どれもわかりやすいものばかりなので、とりたてて説明する必要はないだろう。基本的なファイル操作は置いておくとして、ちょっと便利なGMCの使い方をピックアップしてお届けすることにしよう。

■ディレクトリのサイズを確認する

コンソールで「ls -l」コマンドを実行してもディレクトリのサイズはブロックサイズでしか表示されない。GMCもデフォルトの状態ではそうなっている。コンソールの場合にはduコマンドを使えば表示できるが、GMCにもその機能がある。

［詳細表示］ボタンを押して表示を切り替え、左側のペインからサイズを表示したいディレクトリの親ディレクトリを選択する。たとえば/usr/src/redhat/BUILDのサイズを知りたいなら、/usr/src/redhatを選択しておく。そして、右側のペインで目的のディレクトリを選択し、［ファイル］メニューの［ディレクトリサイズを見る］を実行する。

指定したディレクトリ内にサブディレクトリがあれば、そのサイズも合計して表示してくれる。これは、先ほどの例でいうと「du -s BUILD」としたのと同じ操作だ。もちろん複数のディレクトリを指定して一括して切り替えることも可能（**画面5**）。これは「du -s ＊」と同じ結果になる。

■MIME型を設定してGMCをより多機能にする

MIME型の利点を最大に活かせるのがファイルマネージャであるGMCだ。ビットマップ（BMP）ファイルの例を紹介しながら、MIME型を設定することの利点を見ていこう。

デフォルトでは、拡張子が.bmpのファイルに対してエレクトリックアイズによる表示のみが関連づけられている。BMPファイルをダブルクリックするとエレクトリックアイズが自動的に起動し、そのファイルが表示されるのは［開く］の動作に対してコマンド「ee %f」が関連づけられているからだ。「%f」は実行時に選択されたファイル名に置き換えられる。**画面6**のように指定しておけば、BMPファイルの右クリックメニューから「編集」を選択した際にGIMPが起動し、そのBMPファイルが読み込まれる。

GMCには、デフォルトの状態でも、MIME型を利用した結



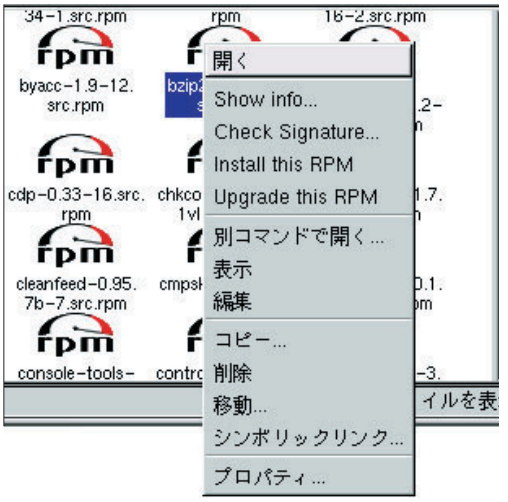
画面4　ミッドナイトコマンダー（CUI）
テキストバージョンはFDライクなファイル操作ツールだ。画面のようにターミナルエミュレータ上でも、標準コンソールでも実行できる。メニューの選択はファンクションキーで行える。



画面5　ミッドナイトコマンダーで複数のディレクトリのサイズを表示



画面6　MIME型の設定
拡張子と実行するファイルを指定するだけの簡単な作業で、GMC上でのファイル操作のバリエーションがぐっと広がる。



画面7　RPMパッケージファイルの右ボタンメニュー
［Show Info］を選択して、RPMの情報やパッケージに含まれるファイルを確認することも可能。

構便利な機能が組み込まれている。そのうちRPMパッケージとMP3ファイルの例を以下で紹介しよう。

■GMCからRPMパッケージをインストールする

GMCでRPMファイルを検索して見つかったら、そのままインストールしてしまおう。GMCでは、通常の右ボタンメニューとは別にRPM用のスペシャルメニューが用意されている。バージョンによってメニュー項目が異なるが、GNOME 1.2.xに付属するGMCには［Install this package］というメニュー項目があり（**画面7**）、選択したパッケージをインストールできるようになっている。

RPM用のメニューには、そのほかにRPM情報やシグネチャの表示、パッケージのアップデートの項目が用意されている。CD-ROMをオートマウントした際にGMCを開くように設定しておけば（64ページを参照）、RPMパッケージのCD-ROMからのインストール手順をずいぶん省力化できるぞ。

■GMCからMP3ファイルを再生する

デフォルトのMIME型の設定で、MP3ファイルのダブルクリックはXMMS（ディストリビューションによってはmpg123）による再生に関連付けられている。また、MP3ファイルを右クリックして表示されるメニューには［Use XMMS to play this］という項目が設定されている。

しかし、どちらの場合も複数ファイルを選択している場合には有効ではない。そこで小技をひとつ。拡張子.m3uファイルにXMMSによる再生を関連付ければいいのだ。.m3uファイルはXMMSで使われるプレイリストファイルで、以下のようにテキスト形式でMP3ファイルのディレクトリパスと名前のリストを書き込んでおけばいい。

```
/home/hogeuser/MySong01.mp3
/home/hogeuser/MySong02.mp3
/home/hogeuser/MySong03.mp3
          :
```

これでリストにしたがって、一度の操作で複数のMP3ファイルを再生できるようになる（**画面8**）。ちょっとしたアイデアだが、けっこう使える技だと思う。いかがだろうか？



画面8　MIME型を利用したプレイリストの再生
プレイリストファイル（.m3u）とXMMSを関連づけておけば（画面右）、ファイルをダブルクリックするだけで複数の曲の再生が可能になる（画面下）。

デスクトップカスタマイズ（KDE編）

KDEは、美しい外観と扱いやすさを兼ね備えた統合デスクトップ環境である。GUIツールキットとして、ノルウェーのTrolltech社が開発したQtを採用している。Qtは、QPLという独自のオープンソースライセンスの下で配布されていたため、Debian GNU/LinuxにはQtとKDEが含まれないといった問題点があった。しかしQtバージョン2.2.0からは、QPLに加えGNU GPL（General Public License）を選択できるようになり、KDEもこれまで以上に積極的な開発が行われると思われる。

本誌12月号でも紹介したように、KDEはメジャーバージョンアップされ、2.0が登場している。だが現状では、ほとんどのディストリビューションでバージョン1.1.2が採用されている状態だ。そこで今回は、KDE 1.1.2／Qt 2.1.1を組み込んでいるKondara MNU/Linux 2000をもとに説明を行っている。カスタマイズを行うKDEコントロールセンターは、KDE 2.0以降でも大幅な変更はなされていないため、操作方法に大差はないだろう。

■KDEコントロールセンター

KDEでは、各種のカスタマイズを「KDEコントロールセンター」からGUIで行えるようになっている（**画面9**）。設定項目は7つのカテゴリーに分類されている。**画面10**に見える8つのカテゴリーのうち、［インフォメーション］は、IRQの割り当てやパーティションの情報など、ハードウェアに関する情報などをチェックするためのカテゴリーだ。コマンドラインで/proc以下のディレクトリを参照する代わりに、利用できる。またそれ以外にも、XサーバやSambaの動作状態など、ソフトウェアに関する情報も得られる。



画面9　KDEコントロールセンター
画面はKondara MNU/Linux 2000のもの。全部で8つのカテゴリーを持ち、広範囲かつ細かな設定が行える。

■パネル関連の設定

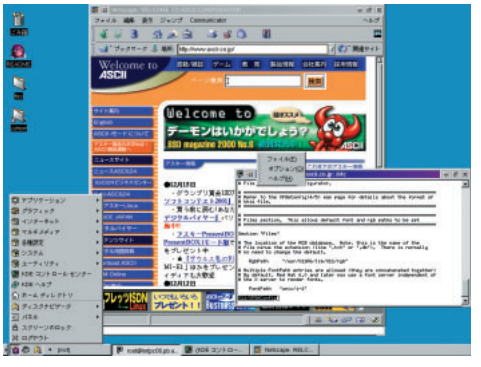
デフォルトのKDEデスクトップ画面では、下部にパネルが、上部にタスクバーが配置されている。パネルには、プログラム起動用のアイコンや仮想画面切り替え用のページャが、タスクバーには起動中のプログラムに対応したボタンが表示されている。機能的にWindows 9xなどと同等といってもよく、便利ではあるのだが、1024×768ピクセル以下の解像度では、実際の表示スペースが狭くなってしまい、たくさんのウィンドウを開くにはちょっとつらいものがある。

大きいディスプレイと高解像度に対応したグラフィックスカードを買うというのが、一番幸せな解決策ではあるのだが、大きいディスプレイは高価だし、かさばるので、そう簡単に買い替えるわけにもいかないものだ。そこで、何とか画面のやりくりをしていくわけだ。そのための設定は、［アプリケーション］カテゴリー内の［パネル］にある。

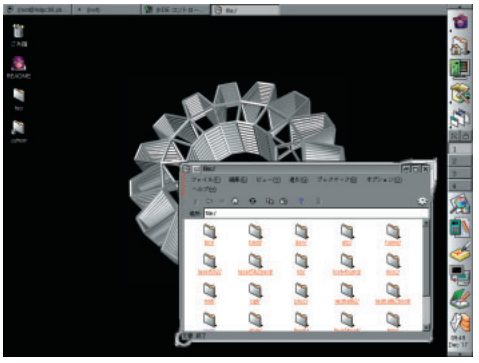
必要のないときは、パネルが見えなくてもかまわないなら、［オプション］タブ内の［自動的に隠す］をチェックしよう。隠れるまでの時間なども設定できるが、長めに設定しておくと、「故障か？」（ソフトウェアだからそんなわけはないのだが）と思ってしまい、かえって気が散るおそれがある。

デフォルトでは上部にあるタスクバーだが、このあたりはウィンドウのタイトルバーがある場所でもあるので、タイトルバーにマウスカーソルを持っていったつもりで、タスクバーが現れたりすると、鬱な気持ちになったりする。そこでタスクバーを下に持っていき、パネルは左か右のボタンを押して格納しておくと、完全に気分はWindowsといった配置のデスクトップができあがる（**画面10**）。タスクバーやパネルの配置、大きさの変更は、［パネル］タブから行えばいい。

Webブラウザやテキストエディタなどは、縦長のウィンドウ



画面10　Windowsライクにした KDE
タスクバーを下部に配置し、Windowsっぽくしてみた。



画面11　タスクバーの配置を変更
タスクバーを縦置きにすると、画面の縦方向が有効に使える。それにこのほうがかっこいい（？）

を開くことが多いが、画面の上下にパネルなどを配置していると、画面サイズのわりにウィンドウが短くなってしまう。気になる人は、パネルを縦に配置しよう（**画面11**）。こうすることで、画面の縦方向を有効に使えるのだ。実は筆者の環境では、OSに関係なく、パネルやタスクバーは右端に立てて置かれている。このほうがかっこいいし、メニューのたぐいは上から降りてくるほうが自然な感じがするからだ。

■パネルにアイコンを追加する

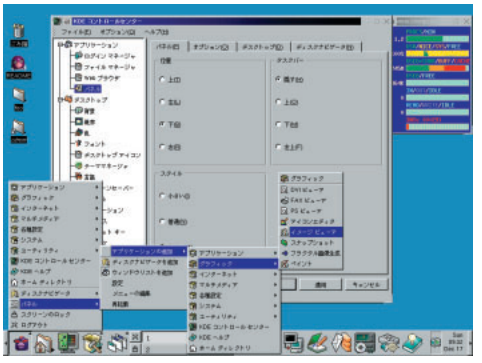
インストールしてすぐの状態でも、最初からいくつかのプログラムがパネルに登録されている。だが、登録されているプログラムを自分がよく使うとは限らないし、登録されていなくてもいつも使うプログラムだってあるはずだ。だから、パネルに登録するプログラムは、自分の好みに合わせてどんどん変更していこう。

GNOMEでは、メニューからパネルにドラッグ＆ドロップすることで登録を行うが、KDEでは、メインメニュー内に［パネル］という項目があり、［アプリケーションの追加］を選ぶことで、メインメニュー内にあるプログラムは、簡単に追加することができる（**画面12**）。もし、後からインストールしたメニューに含まれていないプログラムを追加する場合には、同じく［パネル］の中から［メニューの編集］を選択して、メニューエディタを起動し、追加しよう。

■仮想デスクトップ間の移動

必要なウィンドウが多過ぎて画面に並びきれない場合には、仮想デスクトップを利用するといいだろう。デフォルトでは、4つのデスクトップが用意されている。

別のデスクトップに移動するには、パネル上に表示されているページャをクリックすればいいが、あるデスクトップの端までマウスカーソルを持ってきたら、隣のデスクトップに移動す



画面12　パネルへのプログラムの追加
メインメニューの［パネル］から選んでいけば、簡単に追加することができる。

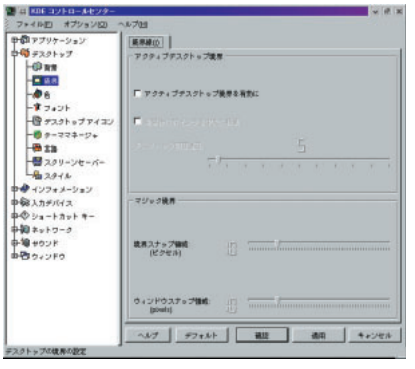
るように設定することも可能だ。［デスクトップ］カテゴリーの［境界］に設定項目がある（**画面13**）。［アクティブデスクトップ境界を有効に］にチェックを入れればいい。アクティブデスクトップといっても、ルートウィンドウが突然Webブラウザになったりはしないので、安心してほしい。

画面の端は、パネルやタスクバーがあるので、マウスカーソルを動かすことが多いが、その時にいきなり隣のデスクトップに移動してしまうとうっとおしいので、デスクトップ切替遅延の値を少し大きめにしておこう。また、隣のデスクトップに移動した後、マウスカーソルを見失わないように、［切替後にポインタを中央に移動］もチェックしておくといいだろう。

■ウィンドウのスナップ移動

KDEでウィンドウをドラッグしていると、隣のウィンドウや画面のふちに吸い付くように感じることがある。これは、ウィンドウのスナップ移動と呼ばれるものだ。ウィンドウを整列させるのにも便利だし、とてもイカす機能といえるだろう。他のOSのいいところを積極的に取り入れる（「真似をする」ではない？）Windowsも次のバージョンできっと導入するに違いない。

どの程度まで近づいたらスナップ移動するかの設定は、同じく［デスクトップ］カテゴリーの［境界］で行う。もちろん、この値を0にすると、スナップ移動しなくなる。



画面13　［デスクトップ］カテゴリの［背景］タブ
仮想デスクトップの切り替えやウィンドウのスナップ移動に関する設定はこのタブで行う。

KFMの機能を満喫すべし

KFMはKDEデスクトップ環境の標準ファイルマネージャであるだけでなく、WebブラウザやFTPのクライアントとしての機能をも備えたすごいヤツなんである。同じファイルマネージャであるGMC（GNOME Midnight Commander）とのGUIツールとしての最も大きな違いは、シングルクリック操作にある。慣れないうちは、やや操作にとまどうことがあるかもしれないが、ひとつのアクションに対する操作が少なくてすむぶん、より軽快なユーザーインターフェイスが実現されている。

GMCと同様に、その機能をいくつかピックアップして紹介していこう。

■Webブラウザとして使う

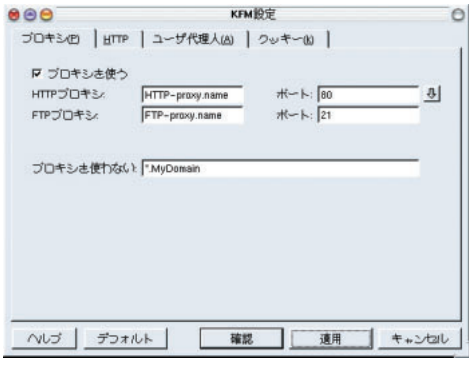
KFMは標準の状態でもHTMLビューアとして機能している。その設定は［ビュー］メニューの［HTMLビュー］にある。このオプションはデフォルトで有効になっているので、そのままHTMLファイルをブラウズすることが可能だ。HTTPプロトコルも解釈するので、実は特に設定を変更しなくてもWebブラウザとして使えるのだ。**画面14**を見てもわかるとおり、フレームを含むページの表示もサポートしている。なお、デフォルトのフォントはKDEのシステムフォントが使用される。



画面14　KFMのHTMLビュー
テーブル、フレームなどの基本的なタグは、きちんと表示できる。ただし、JavaScriptには対応していないようだ。

プロキシサーバを指定することもできる。［オプション］メニューから［ブラウザを設定］を選択すると設定ダイアログが表示される（**画面15**）。［プロキシを使う］をチェックしてプロキシサーバのURLとポートを指定すればOK。画面15からわかるようにFTPプロキシもサポートしている。プロキシサーバを経由せずにアクセスしたいローカル側のホスト名は、［プロキシを使わない］に指定しておこう。

［HTTP］タブでは受け付けるロケールの指定、［クッキー］タブでは文字通りCookieの設定を行える。少しわかりづらいのが［ユーザー代理人］タブだろう。Webサイトによっては、アクセス時にユーザーが使用しているブラウザの種類とバージョンを判断してフレームを含むページの送信を抑止したりしている。先ほど確認したようにKFMはフレームをサポートしているが、Webサイトの側ではそう判断してくれない可能性もある。そこで［サーバ］にWebサイトのURLを、［ログイン］にたとえば「Mozila.4.73」と指定しておけば、指定されたURLのWebサーバはNetscapeを使ってアクセスしていると判断してくれるわけだ。ただ、最近はこうした処理を行っているWebサイトも少なくなっているので、それほど気にする必要はないだろう。



画面15　プロキシサーバの設定
プロキシを使わないホストはワイルドカードでも指定できる。

■イメージビューアとして使う

KFMで指定できるビューには「サムネイルビュー」も用意されている。このオプションを有効にしておけば、イメージファイルのサムネイルが表示されるようになる（**画面16**）。イメージフォーマットとしては、jpeg、xpm、gifなどに対応してい



画面16　KFMのサムネイル表示
メニューバーから［ビュー］－［サムネイル表示］で切り替えられる。

る。KDE 2.xの標準ファイルマネージャKonquerorでは、扱えるイメージフォーマットがさらに増えている（pngなど）。

イメージファイルを右クリックして表示されるポップアップメニューには［イメージビューア］という項目が用意されている。このメニュー項目を選択すると、Kviewが起動し、そのイメージファイルがロードされる。ファイルマネージャ上でプレビューできるので、いちいちファイルを開いて確認しなくて済むのがうれしいところだ。

■KDEアプリケーション以外にMIME型を設定する

KDEでのMIME型（KDEでは「Mimeタイプ」と表示される）の設定は、かなり面倒である。Kviewやkscdなどのようにあらかじめ登録されているKDEアプリケーションはともかく、それ以外のアプリは相当の手順を踏まないと登録できない。ひとつずつ手順を追って説明していこう。例題は「イメージファイルをGNOMEのエレクトリックアイズに関連づける」とする。ここではユーザーのデスクトップ環境のみに反映される「Mimeタイプ」を取り上げるが、システム全体に反映される「広域Mimeタイプ」でも基本的な手順は同じだ。

KFMの［編集］メニューから［Mimeタイプ］を選択する。初期状態ではユーザーごとのMIME型は設定されていないので、ファイルがひとつもないディレクトリ（パスはホームディレクトリの.kde/share/mimelnk/）が表示される。ディレクトリのペインを右クリックしてメニューから［新規作成］－［MIMEタイプ］を選ぶ。MIME型の名前を入力するウィンドウが表示されるので、「images.kdelnk」といった適当な名前を入力しよう（拡張子はデフォルトから変更しないこと）。続いて設定ダイアログの［割り当て］タブに移動し**画面17**のように各項目の値を指定する。複数のパターンを指定する場合は、セミコロン（;）で各パターンを区切って入力する。［標準アプリケーション］の項目については、この段階ではそのままにしておく。

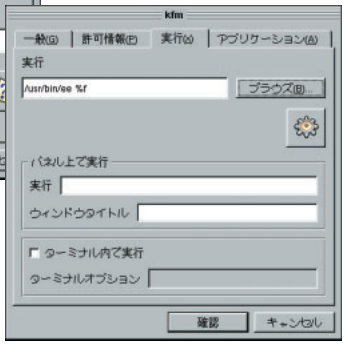
次にアプリケーションの登録が必要になる。KFMの［編集］メニューから、今度は［アプリケーション］を選択する。MIME型の場合と同様に右クリックし、メニューから［新規作成］－［アプリケーション］を選択する。やはり名前の入力を求められる。「ee.kdelnk」とでもしておけばよい。ダイアログで設定が必要なのは［実行］タブと［アプリケーション］タブ。それぞれ**画面18**、**画面19**のように設定しよう。最も重要なのが［アプリケーション］タブの最下段のセクションだ。前の手順で作成したMIME型を右側のリストで選択し、［←］ボタンで左側のリストに登録する。

これで完了といきたいところだが、もうひと手順必要だ。［編集］－［Mimeタイプ］でmimelnkディレクトリに戻って作成したimage.kdelnkをクリックする。表示される設定ダイアログの［割り当て］タブで最下段のドロップダウンリストからエレクトリックアイズを選択すれば、これでようやく完了だ（**画面20**）。横にあるアイコンボタンを押してアイコンの種類を指定することもできる。

では、さっそく効果を試そう。ビットマップファイルなど、先ほど**画面17**で拡張子を関連づけたイメージファイルをKFM上でクリック。見事、エレクトリックアイズが起動すれば大成功だ。うまくいかなかったときは、アプリケーションの実行ファイルのパスや、パターンの指定、MIME型とアプリケーションの関連づけを見直してみよう。



画面17　Minelnkの設定
ここでの設定はユーザー固有のものだが、［Mimeタイプ］には直感的にわかりやすい名前を付けたほうがよいだろう。



画面18　実行ファイルの指定
［ブラウズ］ボタンを押して、プログラムの実行ファイルを探すことも可能だ。アイコンの変更は歯車アイコンをクリック。



画面19　アプリケーションタイプの指定
「日本語で」と書かれているが、別に日本語でなくてもよい。［バイナリパターン］にはパス名なしで実行ファイル名を指定。



画面20　標準アプリの選択
最後に標準アプリケーションを選んでMIME型の設定は完了。紹介した例を参考にして、自分なりのKFM活用法を考案しよう。

綺麗なフォントで デスクトップを飾ろう！

デスクトップカスタマイズ（GNOME）

バージョンアップに伴い、GNOME、KDEなどのデスクトップ環境や、ウィンドウマネージャ、ツールキットの国際化対応が進み、ディストリビューターの側でもそれらに合わせて日本語環境の充実を図ってきた。現在では、どのディストリビューションでもかなりのレベルで日本語環境が整備されている。

とはいえ、たとえばGNOMEの最新バージョンのソースを入手してアップグレードした場合などには、設定が変わってしまって元に戻せないようなこともあるだろう。ここでではフォントの設定に話題を絞って、お気に入りのデスクトップ環境にするための技を紹介しよう。

■GNOMEのフォントを変更する

GNOMEのフォント設定は、メニューやアプリケーションウィンドウ内などはGTK＋、タイトルバーはウィンドウマネージャという2段構えになっている。ユーザーがより多く目にするのはメニューやアプリケーションだろうから、まずはそちらから見ていくことにしよう。

GTK＋のフォント設定は、システム全体のデフォルトが/etc/gtk/gtkrc.ja、ユーザーごとの設定がそのユーザーのホームディレクトリの.gtkrcに保存されている。ユーザーフォントはGNOMEコントロールセンターの［テーマセレクタ］で変更できる。［テーマセレクタ］を選択して表示されるパネルで［ユーザーフォント］の［特別なフォントを使う］を選択して、その下にあるフォント名が表示されたボタンを押すとフォントセレクタが起動する。フォント名と文字コードセットを選んで［了解］ボタンを押し、GNOMEコントロールセンターで［適用］ボタンを押せば完了（**画面21**）。と言いたいところだが、これにはちょっとしたオチがあって、GNOME 1.0.xの場合、この方法では変更が適用されないのであった（確認した範囲ではGNOME 1.2.xはOK）。で、どうするのかというと、ユーザーのホームディレクトリにある.gtkrcファイルを書き換えればいいのだ。.gtkrcに次のような行があるはずだ。

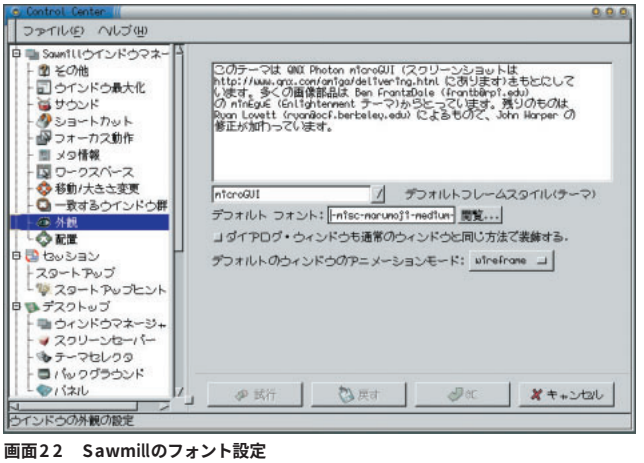
```
font="-misc-fixed-medium-r-normal-*-14-*-*-*-c-*-jisx0208.1983-0"
```

この「font」の部分を「fontset」に書き換えればいいだけ。Xを再起動すれば変更が適用されるはず。/etc/gtk/gtkrc.jaにも同様の「fontset」行があるので、システム全体でフォントを変更したい場合にはその行のフォントの部分を変更すればいい。gtkrc.jaはGUIツールによる変更はできない。

ウィンドウマネージャの側は、代表としてSawmill（Sawfish）を紹介しておこう。GNOMEコントロールセンターの［Sawmillウィンドウマネージャ］から［外観］を選択する。右パネルの［デフォルトで用いられるフォント］の横にあるフォント名のボタンを押すと、フォントセレクタが立ち上がる。あとは、先ほどと同じ要領でフォントを選択して適用すれば完了だ（**画面22**）。



画面21　GTK＋のフォント設定
設定する際に［試写］の部分に表示される文字のフォントが切り替わるようなら、.gtkrcの編集は不要。［適用］ボタンを押すとすぐに設定が反映される。



画面22　Sawmillのフォント設定
フォントセレクタを使わずに［デフォルフォント］の横にあるテキストボックスにXLFD形式でフォントを直接指定することもできる。

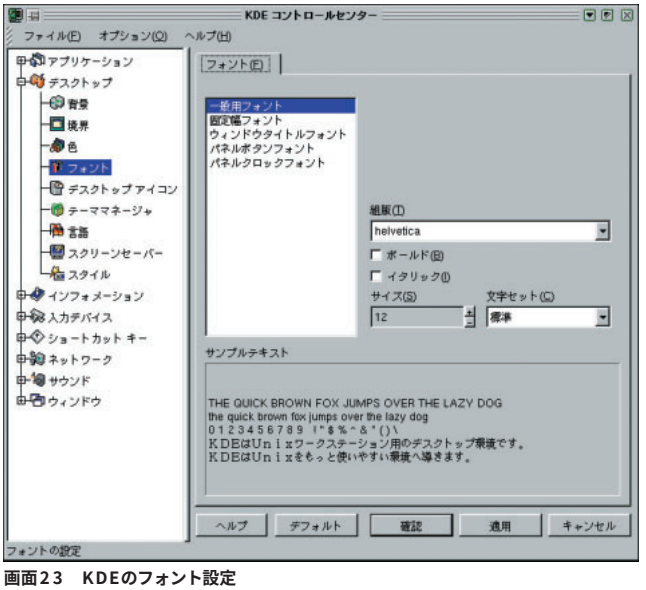
■KDEのフォントを変更する

KDEの場合は、ユーザーのホームディレクトリにある.kdercと.qti18nrcにフォントの設定が保存されている。.kdercが欧文フォント（iso5589-1）、.qti18nrcがそれ以外の文字コードセットのフォント設定ファイルになっている。設定ツールはKDEコントロールセンターだ（**画面23**）。ただし、コントロールセンターを使って変更できるのは.kdercのみだから、話は少しややこしくなる。順を追って説明しよう。

まず、.qti18nrcを編集して日本語フォントと欧文フォントを関連づける。たとえば、こんな感じだ。

```
Helvetica   -wadalab-gothic-*-*-*-jisx0201.1976-0, \
            -wadalab-gothic-*-*-*-jisx0208.1983-0
```

この指定で欧文フォント「Helvetica」に日本語フォントとして和田研のゴシック体が関連づけられる。次にKDEを起動して、KDEコントロールセンターで［デスクトップ］－［フォント］を選択する。［組版］のリストから「Helvetica」を選んで適用すると、欧文フォントがHelvetica、日本語フォントがゴシックになる。欧文フォントを違うものにしたければ、.qti18nrcのHelveticaの部分を書き換えるか、新しく行を追加してそのフォントに関連づける日本語フォントを指定すればいい（もちろん、そのあとKDEコントロールセンターでのフォント変更が必要）。



画面23　KDEのフォント設定
ここで設定できるのは実際には欧文フォントのみ。操作上は日本語フォントを指定することも可能だが、あまり美しい結果は得られないだろう。

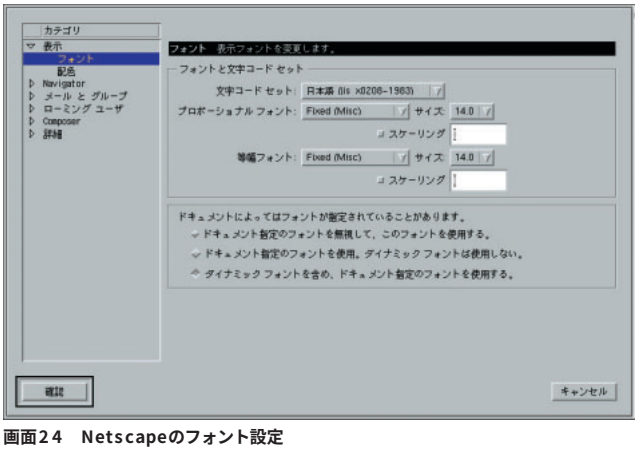
.qti18nrcの欧文フォントの指定（上の例では「Helvetica」）を「＊」にすることで、すべての欧文フォントに1つの日本語フォントを関連付けることもできる。フォント同士のバランスもあるので、いろいろな組み合わせを試してみて最も気に入ったフォントを設定しよう。

■Netscapeのフォントを変更する

Xのアプリケーションで最も長い時間、見つめ続けるのがWebブラウザだろう。デスクトップ環境に依存するものではないが、フォントがらみということで、Netscapeでの設定方法を確認しておこう。

フォントの設定画面は、メニューバーから［編集］－［設定］－［フォント］と順に選択していくと表示される。文字コードセットごとにフォントを設定できるようになっているので、まず［文字コードセット］のリストから「日本語（jisx0208-1983）」を選ぶ。そして［プロポーショナルフォント］と［等幅フォント］のリストから、それぞれフォントを選択すればいい（**画面24**）。プロポーショナルフォントとして固定ピッチフォントを指定しても問題はないので、自分が最も見やすいフォントにするのがgood。

お勧めは「fixed（misc）」の14ポイント。解像度を1280×1024以上にしていると、これではさすがに小さいので、その場合は16ポイントくらいが妥当だ。ホームページのフォントのスケーリングをきちんと表示したい場合は、TrueTypeフォントを選んで、サイズを「0」にして［スケーリング］をオンにしよう。この設定で表示が小さすぎるなら、Xの起動時に「startx - - -dpi 150」と指定するとよい。



画面24　Netscapeのフォント設定
ここで紹介したなかでは最もシンプルに設定可能。ただし、変更した設定はすぐには反映されず、再起動後に有効となる。

知っておきたいデスクトップカスタマイズ（GNOME）

何でもGUIでやろう！

「GUIが使いやすいのは認めるけど、コマンドラインを使わなきゃできないこともあるじゃん」

これは、KDEやGNOMEが進化し、LinuxがWindowsやMacOSなみにGUI化されていく現状を、快く思っていないコマンドライン使いが、口にしたり、心の中で思っていることだ。偉そうに書いているが、少し前までは筆者もそういうコマンドライン使いの一員だった。筆者のLinuxマシンは、X Window Systemが動いてるように見えて、開いているウィンドウは、「ktermが5個」というのが普通だった。

だが、コマンドラインから使うプログラムのGUIフロントエンドや、GUI用に作り直したものが数多くリリースされており、GNOMEやKDEのパッケージに収録されている。コマンドラインなしでも、かなりのところまで作業ができるようになっているのだ。

Linux普及の初期には、「古いマシンでも使える」というのが枕詞のように用いられていた。もちろんそれは今でも正しい。しかし、今や700 MHzのCPUが1万円、256 Mバイトのメモリが1万5000円で買えてしまう時代である。高性能マシンにLinuxをインストールして、GUI三昧で使えば、はっきり言って超快適である。ここではKDEを例に、一度味わうともう抜け出せない、麻薬のような「何でもGUI」の世界を紹介しよう。

■最初は小技から

メニューやパネルに登録していないプログラムは、どうやって起動しているだろうか。そのようなプログラムを起動するために、ターミナルソフトを開いておくのもいいが、ここではAlt＋F2（ファンクションキー）を押してみよう。すると文字列を1行入力できる小さな窓が開くので、ここでプログラムを指定しよう。同じキーバインドは、GNOMEでも有効になっている（コントロールセンターから変更可能）。

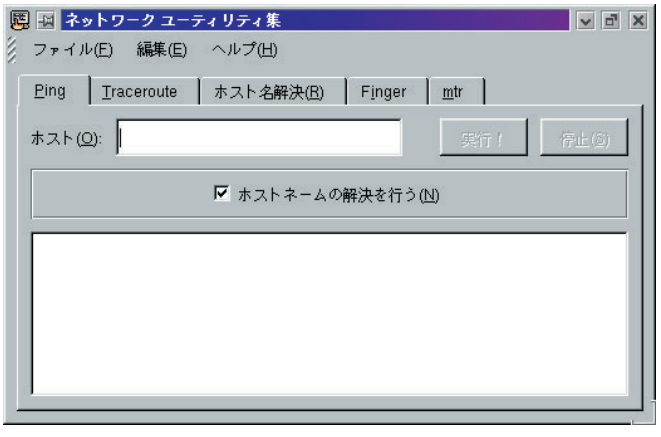
ここで“ls”のような標準出力に結果が出力されるプログラムを実行しても、X上には何も表示されないので気をつけよう。ではどこに表示されているのかというと、startxで起動している場合には、もとのコンソールに出力されてしまうのだ。

■ネットワーク

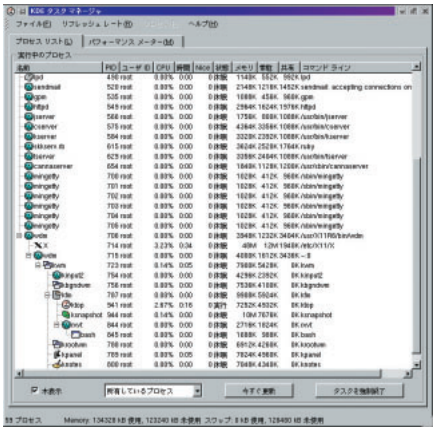
ネットワークは、問題が起きていなければ快適だ。しかし、ひとたびトラブルが起きると、原因を探すために各種のツール類を駆使する必要がある。それらのほとんどは、コマンドラインで利用するものだが、そのGUI版がknuだ（**画面25**）日本語版では、「ネットワークユーティリティ集」と名付けられている。

初期設定では、ping、traceroute、host／nslookup、fingerが使えるようになっている。ホスト名解決（ホスト名とIPアドレスの対応を調べる）には、hostまたはnslookupのどちらかを選択して利用できる。mtrコマンドについては、タブは用意されているが、プログラム本体がインストールされていないので、エラーメッセージが出力される。もっともmtrの機能は、ping、traceroute で代用できるので、それほど困ることもないだろう。

どの機能も、上部のテキストフィールドにホスト名やIPアドレスを入力して、「実行！」ボタンを押せば、下半分に結果が出力されるようになっている。knuが起動している間は、出力結果も保存され、非常に使い勝手がいい。実は、筆者がGUI派



画面25　ネットワークユーティリティ集（knu）
ネットワークのトラブル解決に役立つユーティリティが、ひとまとめになっている。



画面26　プロセスモニタ（ktop）
KDEのtopなので、ktop。そのままの名前だが、使い勝手はいい。

に寝返った（？）理由の半分くらいは、knuがあることを知ったからなのだ。

■プロセスモニタ

起動しているプロセスの確認には、psやtopコマンドが用いられる。topコマンドは、リアルタイムにCPU使用率やプロセスの状態をチェックしたり、シグナルを送ったりできる。しかし、キー1文字のコマンドがなかなか覚えられないので、結局終了の“ｑ”と、シグナルを送る“ｋ”しか使ってなかったりするのだ。

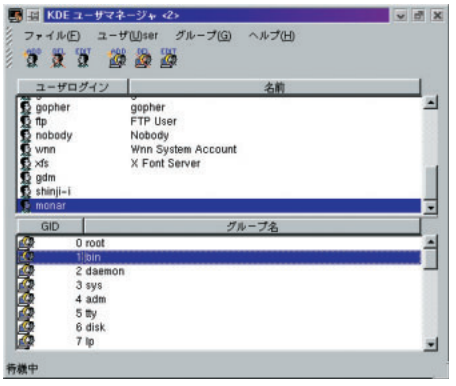
topのKDE版が、ktop（KDEタスクマネージャ）だ（**画面26**）。topと同等の表示が可能なほか、プロセスの親子関係を反映した配置にもできる。ソートする項目の選択は、リスト上部の分類を押せばいいし、シグナルを送るのは、対象となるプロセスの上で右クリックして表示されるメニューからシグナルを選ぶだけだ。

なお、ほとんど同等の機能を持ったkpmというプログラムも用意されている。理由は謎だが、ktopはメニュー内の「システム」カテゴリーに、kpmは「ユーティリティ」に配置されている。

■ユーザー管理

ユーザーやグループの追加、削除、変更は、個別のプログラムは、それぞれ別々のプログラムが用意されている。GUI版のkuser（KDEユーザーマネージャ）は、これらの機能がひとつにまとめられており、使いやすいものに仕上がっている（**画面27**）。Windows NT／2000のユーザーマネージャと同レベルといっていいだろう。

もちろん真のLinuxユーザーは、用途に合わせて、コマンドラインとGUIを使い分けるものだが、今やGUIの比率もかなり高くできるようだ。



画面27　ユーザーマネージャ（kuser）
複数のコマンドラインプログラムの機能がひとまとめになっており、ユーザー管理に関する仕事がすべて行える。

デスクトップに備忘録を書き留める

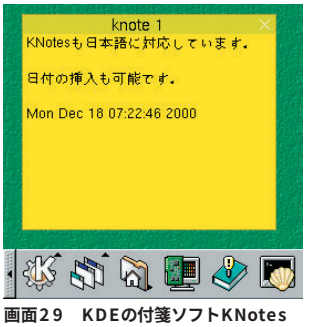
ふだんの生活の中でも「友達にメールの返事を書かなきゃ」とか「○○日までに電話代を払わないと非常にまずい」とか、とにかく忘れちゃいけない事柄が結構ある。付箋にメモしてディスプレイにピトっと貼り付けておくのもひとつの立派な方法だし、GIMPなどでメモをグラフィックにして、壁紙としてデカデカと張り出しておくという荒技も考えられないことはない。

でも、せっかくデスクトップ環境に専用のユーティリティがあるのだから、それを使わない手はない。GNOME 1.2.xにはGNotes！（**画面28**）、KDE 1.1.2にはKNotes（**画面29**）という付箋ソフトが用意されている。

GNotes！は、パネルを右クリックしてメニューから［アプレット］－［ユーティリティ］－［GNotes！］を選択するとパネルに追加される。KNotesはメインメニューの［ユーティリティ］から選択できる。KNnotesのほうが多機能で、日付の挿入、カレンダーの表示（**画面30**）、アラーム（**画面31**）、印刷といったオプションを備えている。



画面28　デスクトップに貼り付けたGNotes



画面29　KDEの付箋ソフトKNotes



画面30　KNotesのカレンダー機能



画面31　設定した時刻に表示されるアラーム

メディアのマウント／アンマウントを自動化する

CD-ROMやフロッピーディスクをドライブに入れたらマウント（mount）、取り出す前にアンマウント（unmount）のコマンドを入力することは、慣れてしまえばなんでもないことだ。でもふだんWindowsやMacOSを使い慣れているユーザーにとっては、「なぜLinuxでは、こんなことが必要なの?」と疑問に思ってしまうポイントでもある。

Linuxでもかっこいいデスクトップ環境を使えるのだから、メディアのマウント／アンマウントも自動でできたほうがいいだろう。

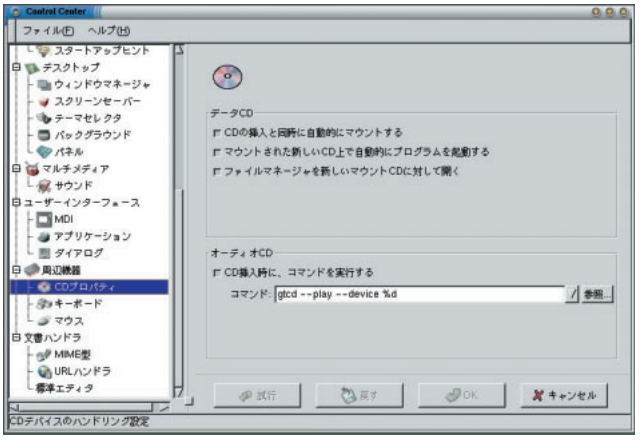
■GNOME用のmagicdev

多くのディストリビューションで、デフォルトのデスクトップ環境となっているGNOMEには、専用のmagicdevという自動マウントデーモンが用意されている。

magicdevは、CD-ROMのドライブへの挿入とボタンを押しての取り出しを認識し、マウント／アンマウントを行う。また、CD-ROMのルートディレクトリに、実行属性を持った“ autorun ”というファイルがある場合、CD-ROM挿入時に、このファイルをプログラムとして自動実行させることができる。Windowsやゲーム専用機のように、CD-ROMを入れたら自動的にゲームが始まるようにすることも可能だ。

さらに音楽CDを挿入した場合には、あらかじめ指定しておいたCDプレーヤソフトを起動し、音楽を再生できる。

magicdevの設定は、GNOMEのコントロールセンターで、左側のリストにある「周辺機器」－「CDプロパティ」を選択すればいい（**画面32**）。これは、Vine Linux 2.1の設定画面だ。ディストリビューションによっては、デフォルトで自動マウントしないように設定されているものもある。



画面32　magicdevの設定画面（Vine Linux 2.1）
コントロールセンター内の表示は、cd-appletというプログラムが行っている。

コントロールセンター内のmagicdevの設定画面は、/usr/bin/cd-appletというプログラムが表示している。さらに、ここで設定した内容は、~/.gnome/magicdevというテキストファイルに記されている。

■autorun

Red Hat系のディストリビューションには、ほかにもautorunというプログラムが用意されている。Red Hat LinuxとLASER5 Linuxは、インストール時に「すべて」や「KDEワークステーション」を選択するとインストールされる。Kondara MNU/LinuxはKondara Zooに、Vine LinuxはVine Plusに含まれている。

CD-ROMのルートディレクトリにある“ autorun ”というファイルを実行する機能や、音楽CDを挿入した場合の自動再生用プレーヤソフトの選択などは、設定ファイルではなくコマンドラインのオプションで指定するようになっている。

■automount

magicdevやautorunと異なり、Linux起動時に起動するように設定されるのが、automountデーモンだ。automountは、直接コマンドラインから起動することもできるが、ほかのデーモンプロセスと同様の制御スクリプトが、/etc/rc.d/init.d/autofsとして用意されているので、これを利用するか、ntsysvコマンドやturboservice（Turbolinux）を用いて起動させるのが一般的だ。

設定ファイルは、メインの/etc/auto.masterとマップファイル（デフォルトでは/etc/auto.misc）からなる。必要ならばマップファイルは、複数指定することもできる。

auto.masterには、マウントポイント（/misc）、マップファイルの名前（auto.misc）を記述し、オプションで自動アンマウントまでの時間などを指定できる。**画面33**の例では、automountでマウントするファイルシステムは、/misc以下に配置され、デバイスとしては、/etc/auto.miscに指定したCD-ROMとフロッピーディスクが利用できる。さらに60秒間アクセスがなければ、自動的にアンマウントされるようになっている。

マップファイル（auto.misc）には、自動マウントしたいデバイスのキー、ファイルタイプなどのオプション、そしてデバイスファイル名を1行ずつ記す。それぞれのデバイスは、auto.masterで指定したマウントポイント下の、サブディレクトリとしてアクセスできる。

画面33の例では、/dev/cdrom（通常の構成では/dev/hdcへのリンクになっていることが多い）で表されるドライブにCD-ROMを入れ、コマンドラインから、

```
$ ls /misc/cd
```

とすることで、自動的にCD-ROMがマウントされ、リストが表示される。ファイルマネージャで/misc/cdディレクトリを指定した場合も同様だ。

■どれを使うべき？

magicdev、autorun、automountは、ほぼ同等の機能を提供するプログラムなので、混在させて使うと、CD-ROMドライブの状態を正しく認識できず、CD-ROMが取り出せなくなったり、アンマウントできなくなったりすることがある。

GNOME、KDEの環境から使うことが多いなら、それぞれmagicdev、autorunを使い、automountデーモンは止めておくのがいいだろう。automountは、NFSやSMBファイルシステムを利用して他のマシンのリソースを自動マウントする場合だけ用いるのが適切だ。

```
[bash]$ cat -n /etc/auto.master
 1: # $Id: auto.master,v 1.2 1997/10/06 21:52:03 hpa Exp $
 2: # Sample auto.master file
 3: # Format of this file:
 4: # mountpoint map options
 5: # For details of the format look at autofs(8).
 6: /misc   /etc/auto.misc  --timeout 60

[bash]$ cat -n /etc/auto.misc
 1: # $Id: auto.misc,v 1.2 1997/10/06 21:52:04 hpa Exp $
 2: # This is an automounter map and it has the following format
 3: # key [ -mount-options-separated-by-comma ] location
 4: # Details may be found in the autofs(5) manpage
 5: cd              -fstype=iso9660,ro      :/dev/cdrom
 6: floppy          -fstype=auto            :/dev/fd0
```

画面33　auto.masterとマップファイルの例
auto.masterには、マウントポイント、マップファイル、オプションを指定する。マップファイルでは、自動マウントするデバイスのキー、ファイルシステム、デバイス名を指定する。

デスクトップ環境のお引越し、隠しファイルがいっぱい！

ディストリビューションを変えた際、せっかくカスタマイズしたデスクトップ環境の移行方法がわからないがために、もう一度、初めから設定をやり直したという経験はないだろうか。ひとつ、デスクトップ環境の引越しについて考えてみよう。

まず別のPCや別のディストリビューションへの移行について。このとき、まず第一に考慮しなくてはならないのが、GNOMEやKDEのバージョン。1.1.1と1.1.2といったリビジョン以下の違いであれば設定項目や設定ファイルの形式に違いはないと考えていいが、KDEのテーマのようにリビジョンの違いで互換性のないものもある。移行元と移行先で、まったく同じバージョンであることが望ましいのは言うまでもない。設定されているテーマやフォントが移行先にインストールされているかどうか、といった点も確認しておこう。また、移行元でシステムのデフォルト設定（/etc/gtk/gtkrc.jaなど）を変更している場合、ホームディレクトリだけでなくそれらの設定ファイルも移行する必要があることに注意しよう。

同じシステム内での移行には、それほど問題点はないはずだ。というわけで、あとはどのファイルをコピーするのか、という判断がつけば移行できる。GNOMEの場合は.gtkrcと、.gnome/、.gnome-help-browser/、.gnome_private/以下のファイルとディレクトリ、.sawmill（.sawfish/）、.enlightenment/、e-confなどウィンドウマネージャ関連のデイレクトリ、そして.gnome-desktop/のハードウェアに関連していないファイルが対象となる。KDEでは、.kdercと.qti18nrc、.kde/と以下のファイルとディレクトリ、Desktop/のハードウェアに関連していないファイルを移行すればよい。.netscape/や.xmmsなどのアプリケーションの設定ファイルも忘れずに。特に.netscape/にはブックマークや履歴、アドレス帳などのファイルが含まれているので移行したい。ただし、キャッシュファイルの作成先パスは、パーミッションの問題があるため、設定し直す必要があるかもしれない。コピーが完了したら、必ずファイルとディレクトリの所有者を変更すること。コマンドは「chown -hv ユーザー.グループ ファイル名」または「chown -hRv ユーザー.グループ ディレクトリ名」だ。

おさえておきたい中級技（X Window System編）

クリクリ動かす

Xにおける手抜き操作のススメ

基本的な技はマスターしたんだけど、もうちょっと突っ込んだ技も使ってみたいとお考えなら、このセクションに目を通しておこう。古くから伝わる技や、比較的最近のハードウェアであるホイールマウスの使い方まで、バラエティに富んだ技を紹介している。長く険しいX Window Systemマスターへの道の道標として利用してほしい。

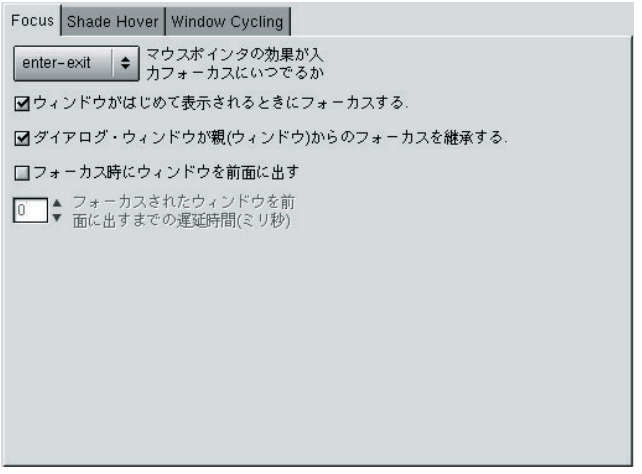
ウィンドウフォーカス

ウィンドウシステムを備えたOSでは、複数のウィンドウを開いて、作業を並行して行うことができるが、そのうちキーボードやマウスの入力を受け取るウィンドウは「フォーカスしている」とか「アクティブな」などと表現する。

X Window Systemの場合、フォーカスのしかたを決めるのは、ウィンドウマネージャの役割である。一般的に、ディストリビューションに含まれているウィンドウマネージャは、MacOSやWindowsと同じように、最も手前のウィンドウにフォーカスしているように設定されている場合が多い。

GNOMEコントロールセンターの「Sawmillウィンドウマネージャ」（Sawmillという名前が古いが気にしない）カテゴリーの「フォーカス動作」という項目を選択すると、**画面1**のような設定項目が表示される。デフォルトでは、マウスをウィンドウ上でクリックするとフォーカスが移動するようになっているが、リストから「enter-exit」か「enter-only」を選択すると、カーソルがウィンドウの上に乗っただけで、フォーカスが渡るようになる。いちいちクリックしないでも済むので、こちらを好むユーザーも多い。その場合は、あわせて［フォーカス時にウィンドウを前面に出す］のチェックを外しておいたほうがいい。そうしないと、マウスが動いてしまったときに、ウィンドウが勝手に手前に出てきてうっとおしいからだ。

フォーカスがあるウィンドウを最前面に持ってきたい場合は、ウィンドウ内でAlt＋右クリックすればいい。



画面1　Sawfishでのウィンドウフォーカスの設定
いくつかのオプションは、まとめて変更したほうが、快適な操作環境が得られる。

Cannaを使う

「もう使ってるよ」という方は次のトピックへ。

Cannaは、ほとんどすべての国産ディストリビューションに含まれており、Cannaの操作を会得していればLinuxで日本語入力に困ることはないはず。**表1**にCannaにおける日本語入力のキー操作をまとめておくので、これを参考に基本的な操作をマスターしよう。

■カタカナを入力する

表1の中でCtrl＋nは「字種の切り替え」となっている。たとえば「あすきー」と入力してCtrl＋nを押すと、アスキー（全角カタカナ）→ｱｽｷｰ（半角カタカナ）→ａｓｕｋｉ－（全角英数字）→asuki-（半角英数字）と文字の種類が切り替わる。つまり、Ctrl＋nを1回押すことは「カタカナ変換」ということになるのだ。なお、Ctrl＋nはカーソルキーの上（↑）で代替可能だ。

■半角英数字を入力する

カタカナ入力の変則技として「半角英数字変換」がある。「Linux」と入力したいときに、この順でキーを押すと「Lいぬx」となる。Ctrl＋nキーを2回押してもいいが、キータッチはなるべく少なくしたい。Ctrl＋pを押すとCtrl＋nの逆順に文字種が切り替わる。つまりCtrl＋pキー1回で半角英数字変換となる。Ctrl＋pはカーソルキーの下（↓）で代替可能だ。

■記号を入力する

「○」を入力する場合を例に説明しよう。通常の入力モードで○の漢字コード「312b」を入力し、Ctrl＋yキーを押す。

日本語入力の開始／終了	Ctrl＋スペース
かな漢字変換／変換候補の表示	スペース
字種の切り替え	Ctrl＋n／Ctrl＋p
文字コードからの直接変換	Ctrl＋y
変換する文節を縮める	Ctrl＋o（Shift＋←）
変換する文節を広げる	Ctrl＋i（Shift＋→）
変換の確定	Enter
記号入力モードへの移行	Insert
拡張モードへの移行	Home
通常モードへの復帰	Ctrl＋h

表1　Cannaの基本的な操作（kinput2）

これだけである。手順は簡単だが、ふつうの人は漢字コードを記憶していないと思うので、そこが問題。より便利なのが記号入力モードを使う方法だ。

Insertキーを押すと記号入力モードに切り替わる（**画面2**）。表示されるリストから「○」を選んでEnterキーを押せばOK。こちらも手順は簡単だ。

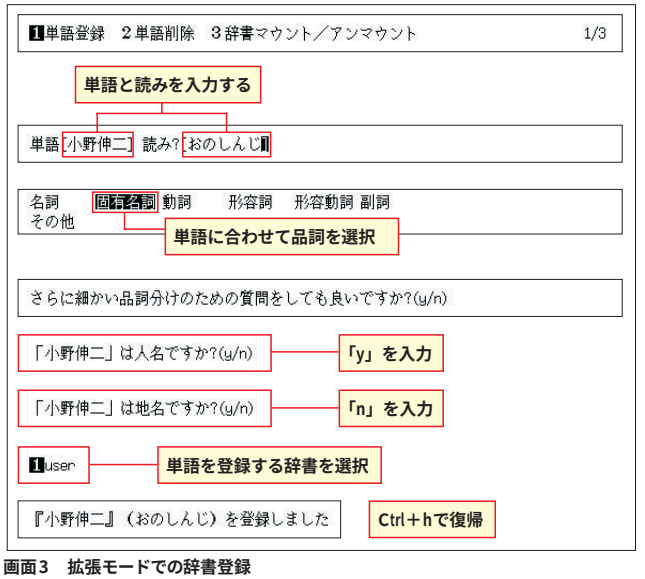
■辞書を鍛える

Cannaは辞書登録機能も備えている。この機能は拡張モードからアクセスできる。通常の入力モードでHomeキーを押して拡張モードに移行し、メニューから［単語登録］を選択。あとは指示にしたがって対話的に進めていく。詳しい手順は画面2を参考にしてほしい。**画面3**の手順で登録した単語はユーザー辞書に登録される。

辞書のマウント／アンマウントも拡張モードメニューの［単語登録］から行うことができる。デフォルトでインストールされている辞書には、地名辞書、部首引き辞書などがある。フリーで配布されている辞書としては、熊田信義氏が作成した郵便番号辞書がお勧め。興味のある方は熊田氏のWebサイトを一度訪ねてみよう（http://member.nifty.ne.jp/kumada/）。



画面2　Cannaの記号入力モード



画面3　拡張モードでの辞書登録

ホイールマウスをクリクリ動かす

Windowsユーザーの間では、かなり普及しているホイールマウス（筆者は個人的にクリクリマウスと呼んでいる）。Webブラウザでスクロールするときなど、とても重宝する。ぜひともXで使えるようにしたいアイテムのひとつだ。

XFree86 3.3.2以上なら、XF86Configの設定でホイールの動きをイベントとして割り当てるだけで簡単にホイールマウスを使うことができる。手持ちのマウスがMicrosoftのIntelliMouseでPS/2コネクタ接続ならば、/etc/X11/XF86Configの「Pointer」セクションを**リスト1**のように編集すればOK。ロジクール（米Logitech）のMouseMan+の設定も、コメントにして示してあるので参考にしてほしい。

この設定でマウスの左ボタンが「ボタン1」、中ボタンが「ボタン2」、右ボタンが「ボタン3」、ホイールの上への回転が「ボタン4」、下への回転が「ボタン5」として認識されるようになる。GNOMEには、このホイールイベントに対するハンドラが用意されているので、gEditやファイルマネージャ（GMC）などのGNOMEアプリケーションは、これだけで画面のスクロールにホイールが使えるようになる。

しかしなんといっても、Webブラウザで使うのが一番便利だ。Netscape Navigatorで使用するには、マウスイベントをNavigatorの各機能に関連づけるためのさらなる設定が必要になる。ホームディレクトリの.Xdefaultsに**リスト2**の内容を追加しよう。これでWebのブラウズでホイール機能が使えるようになるぞ。Navigatorのように、デフォルトではホイールのイベントに対応していないアプリケーションの場合、個別の設定が必要になる。ほかのアプリでのホイール機能の設定については、http://www-sop.inria.fr/koala/colas/mouse-wheel-scroll/に詳しく解説されている。お気に入りのアプリが載っていたら、さっそく設定してクリクリ感を楽しもう。

▼リスト1　XF86Configでのホイールマウスの設定

```
Section "Pointer"
        ...
    Protocol "imps/2"          # 標準の「PS/2」から変更する
#   Protocol "MouseManPlusPS/2"  # MouseMan+用の設定はこちら
    ZAxisMapping 4 5           # ホイールの操作をボタン4、5に割り当てる
        ...
EndSection
```

▼リスト1　XF86Configでのホイールマウスの設定

```
!## NETSCAPE
Netscape*drawingArea.translations: #replace  \
  <Btn1Down>: ArmLink()  \n\
  <Btn2Down>: ArmLink()  \n\
  ~Shift<Btn1Up>: ActivateLink()  \n\
  ~Shift<Btn2Up>: ActivateLink(new-window)  \
                   DisarmLink()  \n\
  Shift<Btn1Up>: ActivateLink(save-only)  \
                  DisarmLink()  \n\
  Shift<Btn2Up>: ActivateLink(save-only)  \
                  DisarmLink()  \n\
  <Btn1Motion>: DisarmLinkIfMoved()  \n\
  <Btn2Motion>: DisarmLinkIfMoved()  \n\
  <Btn3Motion>: DisarmLinkIfMoved()  \n\
  <Motion>: DescribeLink()  \n\
  <Btn3Down>: xfeDoPopup()  \n\
  <Btn3Up>: ActivatePopup()  \n\
  Ctrl<Btn4Down>: PageUp()  \n\
  Ctrl<Btn5Down>: PageDown()  \n\
  Shift<Btn4Down>: LineUp()  \n\
  Shift<Btn5Down>: LineDown()  \n\
  None<Btn4Down>: LineUp()LineUp()LineUp()LineUp()LineUp()LineUp() \n\
  None<Btn5Down>: LineDown()LineDown()LineDown()LineDown()LineDown()LineDown() \n\
  Alt<Btn4Down>: xfeDoCommand(forward)  \n\
  Alt<Btn5Down>: xfeDoCommand(back)  \n
```

ホイールアップ＋Ctrlキーでページ単位のスクロールアップ
ホイールダウン＋Ctrlキーでページ単位のスクロールダウン
ホイールアップ＋Shiftキーで1行ずつのスクロールアップ
ホイールダウン＋Shiftキーで1行ずつのスクロールダウン
ホイールアップで6行単位のスクロールアップ
ホイールダウンで6行単位のスクロールダウン
ホイールアップ＋Altキーで「進む」
ホイールダウン＋Altキーで「戻る」

Xにおける手抜き操作のススメ

X Window Systemには、マウスによる伝統的なコピー＆ペースト方法や、軽快なウィンドウ操作を実現してくれるショートカットキーなど、多くの「手抜き技」が存在する。ここで言う手抜きとは、ネガティブなものではなく、手順を省略することで操作の簡略化を目的とした合理的なものである。

■マウスによるコピー＆ペースト

ベテランのUNIX使いであれば、すべてのX環境で共通に使えるマウスによる「コピペ」技をご存じだろう。X初級者にも、古（いにしえ）より伝わるこの技を伝授しよう。

まず、アプリケーションウィンドウ内に表示されたコピー元となる文字列をマウスの左ボタンを「ズズズー」とドラッグして反転表示させる。次にコピー先にマウスポインタを移動し、おもむろに中ボタンをクリック。すると、あら不思議。先ほど選択した文字列がペーストされるではないか！　念のために言っておくと、コピー先はテキスト入力が可能なウィンドウ領域でなければならない。タイトルバーなどにペーストしようとして失敗に終わった諸君は、すみやかに入力可能領域（テキストエディタなどがよろしかろう）で再度試みてほしい。

ちと表現力が足りないので、理解できなかったやもしれぬ。このページの下段で図解しておいた（**図1**）ので、参考にされたい。この技を習得しておけば、Webブラウザに表示されたホームページのテキストをエディタに取り込むであるとか、あるいはターミナルエミュレータにおいて表示された長いファイルパスをコマンドの引数としてコピーするとかいった操作がチョー簡単に行える。必ずやあなたの役に立つはずである。

■ショートカット

ショートカットキーはウィンドウマネージャごとに異なり、またカスタマイズ可能な設定項目なので、すべての環境で使えるわけではないが、多くの環境で使うことができる汎用的なものを覚えておくのは無駄にはならないないだろう。

たとえば、通常はタイトルバーをドラッグしてウィンドウ移動するが、Altキーを押しながら左ボタンでドラッグすれば、任意の場所にマウスポインタを置いた状態でそのウィンドウを移動できる。このショットカットキー（純粋には「キー」ではないが）は、タイトルバーがほかのウィンドウで隠れてしまっている場合などに有効だ。また、WindowsでもおなじみのAlt＋Tabキーによるウィンドウの切り替えも、多くのウィンドウマネージャでサポートされている。

各ウィンドウマネージャでデフォルト設定されているものには、Altキー＋マウスボタン（左／右／中）というコンビネーションのものが多くある。いろいろ試してみると新たな技を発見できるかもしれない。もちろん、設定ツールを使って独自のショートカットを登録してもいい。アイデアしだいでマウス操作のいらない究極のショートカットキー環境を作り上げることだって可能なのだ。

ごく一部ではあるが、すぐに役に立ちそうなショートカットキーを**表1**にまとめてある。偉大なるショートカットマスターへの道はここから始まる。

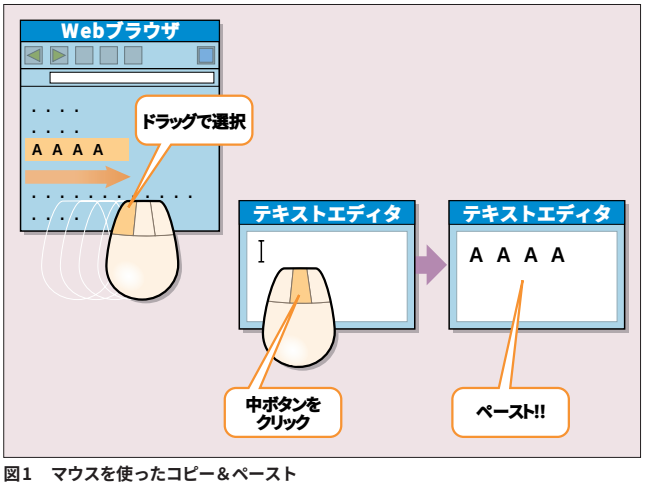


図1　マウスを使ったコピー＆ペースト

操作	キー＆マウスコンビネーション		
	Sawfisht	kwm	Enlightenment
ウィンドウの任意の場所をポイントして移動	Alt＋左ドラッグ	Alt＋左ドラッグ	Alt＋左ドラッグ
アクティブウィンドウの切り替え	Alt＋Tab	Alt＋Tab	Alt＋Tab
タイトルバーをドラッグしてウィンドウのサイズを変更	中ドラッグ	Alt＋右ドラッグ	A＋中ドラッグ
背後にあるウィンドウを前面に表示	Alt＋右クリック	—	Ctrl＋Alt＋↑
アクティブウィンドウのクローズ	—	Alt＋F4	Ctrl＋Alt＋x
ワークスペース領域の切り替え	—	Ctrl＋F1〜F8	Alt＋F1〜F8

表2　代表的なショートカットの例

Linuxのあんな技こんな技（コマンド編）

最近では、Linuxの起動とともにXが立ち上がり、すべての操作はX環境で行うというユーザーも多い。しかし、マシンの環境によってはXが起動しない場合もあるし、サーバマシンなどではXがインストールされていないことも多い。そこで、ここでは原点に立ち返り、コンソールを使ううえで役に立つ（かもしれない）Tipsや、デーモン／日常管理に関する小技を紹介しよう。肩の力を抜いて、のんびりと読んでほしい。いつに日か何かの役に立つかもしれない。

コンソールでマウスを使う

ほとんどのディストリビューションで、インストール直後からコンソール用のマウスサーバgpmが動作している。gpmは、マウスを使った文字列のコピー＆ペースト機能を提供する。

コンソールを表示した状態でマウスを動かすと、画面に四角いマウスポインタが現れるだろう。マウスの左ボタンを押しながらコピーしたい文字列をマークし、中ボタン（2ボタンマウスなら右ボタン）でペーストすることができる。gpmのエラいところは、仮想コンソールをまたいだコピー＆ペーストができることだ。左ボタンでコピーしたら、Alt＋ファンクションキーで仮想コンソールを切り替え、中ボタンでペーストすればよい。余談だが、仮想コンソールの切り替えにはAlt＋←→キーを利用することも可能だ。

このほか、gpmパッケージには“ gpm-root ”というプログラムが付属する。このプログラムを起動しておくと、Ctrl＋マウスボタンでポップアップメニューが表示される。このメニューから、仮想コンソールを切り替えたり、コマンドを実行させることができるのだ（**画面1**）。

メニューの内容は/etc/gpm-root.confファイルで設定できる。初期状態では、左ボタンと右ボタンで仮想コンソールの切り替え、中ボタンでシステムの状態表示をするようになっている。gpm-root.confの最後のほうには、システムのシャットダウンやリブートという項目まで用意されている（危険なのでコメントアウトされているが）。



画面1　ポップアップメニュー
左は仮想コンソールの切り替えメニュー。右はシステムの状態表示。到着したメール一覧も表示可能だ。

知っておきたいデスクトップ環境を活かすちょっといい技
広いコンソールを使う

PC-9800シリーズのDOSでは「30行計画」というフリーソフトによってテキスト画面を広くして使うことができた（5行ほど増えるだけだが、当時は非常に便利だったのだ）。また、DOS/Vでは、Hi-TextやV-Textによって、高解像度の画面による利用が可能だった。実は、わがLinuxでも同じようなことができるのだ。

■konで800×600ドット環境を使う

日本語コンソールを実現するkonは、800×600ピクセルの画面にも対応している。konを利用できる環境であれば、次のようにしてkonを起動してみよう。

```
$ kon Stealth
```

運が良ければ、これだけで800×600ピクセルの日本語コンソール画面になるはずだ。konの設定は、/etc/kon.cfgで行う。このファイルを開いてみると**リスト1**のような部分が見つかるだろう。これは、ディスプレイの設定を行う部分で、「Stealth」というエントリは800×600ピクセル環境の定義を行っている。konは起動時にディスプレイ定義のエントリをオプションとして指定できるので、先ほどの例ではこのエントリが使われているのだ。ちなみに、「Stealth」というのは、S3 86C911チップを搭載したDiamondのグラフィックスカードStealth VRAMのことだと思われる。

▼リスト1　kon.cfgの内容（一部）

```
                :
                :
Stealth:
        VGA
        800 864 1016 1064 600 606 608 632
        2
        99 36
                :
                :
VGA:Normal
        VGA
        640 680 768 800 480 491 493 525
        1
        79 29
                :
                :
```

さて、話を戻そう。konの起動時、オプションに何も指定しないと「Normal」というエントリが使用される。エントリ名は、：（コロン）で区切って別名も指定できる（printcapなどと同様だ）ので、「VGA:Normal」という640×480ピクセル環境のエントリが使用される。

これらのエントリには、ドライバ名（VGAなど）、画面制御パラメータ、クロック番号、テキスト画面の広さを指定する。各項目の詳細は、konのmanページを参照してほしい。

■フレームバッファコンソールで広大なコンソールを

カーネルがフレームバッファ（fb）コンソールをサポートしていれば、**画面2**のような広いコンソール画面を手軽に利用できる。KondaraやSlackware 7.xには、fbコンソールをサポートするカーネルが含まれているが、残念ながら、ほかのほとんどのディストリビューションではカーネルの再構築が必要だ。

カーネルを準備したら、/etc/lilo.confで「VGA＝VESAモード番号」を指定する。代表的なVESAモード番号は**表1**のとおりだ。

VESAモード番号	画面	VESAモード番号	画面
791	1024×768×64色	771	800×600×256色
790	1024×768×32k色	785	640×480×64k色
773	1024×768×256色	784	640×480×32k色
788	800×600×64k色	769	640×480×256色
787	800×600×32k色		

表1　代表的なVESAモード番号



画面2　1024×768ドットのコンソール画面
シンボリックリンクがたくさん張られていても安心してls -lを実行できる。う〜ん、快適。

リダイレクトやパイプを使いこなす

シェルから対話的に利用するUNIXのソフトウェアは、「標準入力」という入力の受け取り手段と、「標準出力」、「標準エラー出力」という出力の生成手段を持っている。通常、標準入力はキーボードに、標準出力と標準エラー出力は端末画面に設定されているので、これらのソフトウェアはキーボードからの入力を処理して、端末の画面に結果を出力する（**図1**）。たとえば、入力から出力へデータをコピーするcatコマンドをオプションなしで起動すると、キーボードからの入力を待ち、入力があるとそれを画面に表示（出力）する。

この入力もとや出力先を切り替えるのがリダイレクション（リダイレクト）やパイプライン（パイプ）という仕組みだ。コマンドに対して、ファイルからデータを入力したり、結果をファイルへ出力するのにはリダイレクトを使う。コマンドの出力をほかのコマンドに渡す場合はパイプを利用する。ここでは、bashでリダイレクトとパイプを使う方法を説明しよう。

■出力のリダイレクト

コマンドの結果をファイルに保存したい場合は、標準出力を画面ではなく、ファイルに切り替えればよい。そのためには、コマンドのあとに出力のリダイレクト記号「>」を付け、保存したいファイル名を指定する。たとえば、次のようにするとキーボード（標準入力）からの入力をhoge.txtというファイルに出力する。

```
$ cat > hoge.txt
```

これを実行するとキーボードからの入力待ちになるので、文字を入力すればそれがhoge.txtに保存される（入力の終了はCtrl＋Dキー）。これを再び実行すると、hoge.txtの内容が消去され、新たに入力された文字が保存されるが、「>」の代わりに、「>>」を使うと、内容は消去されず、hoge.txtの末尾に文字が追加される。

さて、出力には標準エラー出力もある。なぜ出力が2系統用意されているのだろうか。これは、その名のとおり、エラーメッセージを出力するのに使われる。標準出力をファイルにリダイレクトすると、その出力内容は画面に表示されなくなる。しかし、エラーメッセージは画面に表示されたほうが便利だ。そこで、エラーメッセージは標準エラー出力に出力するのだ。実験してみよう。aru-fileというファイルだけが存在するディレクトリで、次のようにlsコマンドを実行する。

```
$ ls aru-file nai-file
ls: nai-file: No such file or directory
aru-file
```

nai-fileは存在しないのでエラーメッセージが表示され、そのあとにaru-fileがリストされた。どちらも画面に表示されているが、エラーメッセージは標準エラー出力に、ファイルリストは標準出力にそれぞれ出力されている。これは、標準出力をファイルに切り替えるとよくわかる。

```
$ ls aru-file nai-file > list.txt
ls: nai-file: No such file or directory
```

今度はエラーメッセージだけが表示された。標準出力に出力されるファイルリストは、list.txtというファイルに書き込まれ、標準エラー出力だけが画面に表示されたのだ。

標準エラー出力をリダイレクトすることも可能だ。それには、

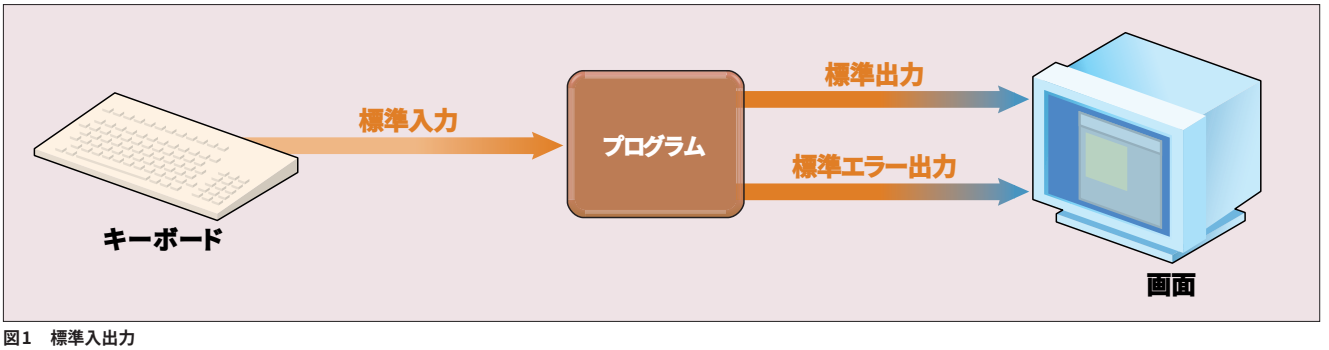


図1　標準入出力

リダイレクト記号「2>」を使う。次のようにすれば、標準出力と標準エラー出力を別々のファイルに保存できる。

```
$ ls aru-file nai-file > list.txt 2> error.txt
```

ファイルリストはlist.txtに、エラーメッセージはerror.txtに書き込まれる。「2>」の2はファイルディスクリプタというもので、標準入力は0、標準出力は1、標準エラー出力は2と決められている（したがって、「>」は「1>」と等価だ）。

では、標準出力と標準エラー出力の両方を1つのファイルに保存するにはどうすればいいだろうか。それには、リダイレクト記号「&>」を使う。

```
$ ls aru-file nai-file &> zenbu.txt
```

このほかに、「>&」というリダイレクト記号を使う方法もある。これは、指定したファイルディスクリプタの出力を別のファイルディスクリプタの出力にコピーする記号だ。たとえば、「2>&1」とすると、標準エラー出力（ファイルディスクリプタ2）を標準出力（ファイルディスクリプタ1）にコピーすることを意味する。実際の例を見たほうがわかりやすい。

```
$ ls aru-file nai-file > zenbu.txt 2>&1
```

このようにすると、標準出力はzenbu.txtに書き込まれ、「2>&1」の指定により標準エラー出力は標準出力と同じところへコピーされる。この結果、標準出力と標準エラー出力の両方がzenbu.txtに書き込まれる。

```
$ ls aru-file nai-file 2> zenbu.txt >&2
```

もちろん、このようにしても同じ結果が得られる。

■入力のリダイレクト

標準入力に対し、キーボードの代わりにファイルからデータを入力する場合は、入力のリダイレクト記号「<」を使う。たとえば、漢字コード変換コマンドnkfを使い、シフトJISのテキストファイルの内容をEUCコードに変換して画面に表示するには、

```
$ nkf -e < sjis-file.txt
```

のようにする。-eは、EUCコードに変換するオプションだ。この出力をファイルに保存するなら、

```
$ nkf -e < sjis-file.txt > euc-file.txt
```

とすればよい（実は、nkfは引数に読み込むファイル名を指定できるので、「<」を使わなくてもよい。引数がない場合は標準入力から読み込むので「<」が使えるのだ）。

■コマンドをパイプでつなぐ

あるコマンドの出力をほかのコマンドに渡したいときはパイプを使う。たとえば、Linux起動時のシステム情報を表示するdmesgコマンドのように、出力が1画面に収まらず、スクロールしてしまう場合は、パイプ記号「|」でlessなどのページャにつなぐと便利だ。

```
$ dmesg | less
```

こうすることで、dmesgが標準出力に対して出力する内容をlessが受け取り、内容を表示する。このほか、3つ以上のコマンドをパイプでつなぐことも可能だし、最終的な出力をリダイレクトすることもできる。

```
$ cat hoge.txt | grep moge | sort > kekka.txt
```

標準出力と標準エラー出力の両方をほかのコマンドに渡す場合は次のようにする。

```
$ ls aru-file nai-file 2>&1 | less
```

リダイレクトと同じように「&|」や「2|」ができると便利かもしれないが、残念ながらできない。

また、標準入力から受け取った内容を画面に表示するとともに、指定したファイルに書き込むteeコマンドも便利だ。

```
$ ls -l | tee ls-l.txt
```

このようにすれば、ls -lの結果を画面に表示し、同時にls-l.txtというファイルにも書き込む。

早朝ディスクアクセスのナゾ

午前4時を過ぎたあたりでLinuxマシンが激しくディスクアクセスを始める。これは、決まった時間になるとプログラムを動かすcronデーモンの仕業だ。多くのディストリビューションでは、システムを維持するための作業をこの時間に行うよう設定されている。cronの設定は/etc/crontabと/etc/cron＊以下のファイルで行われている。

さて、ここで注目したいのは、/etc/cron.daily/slocate.cronというファイルだ。これはslocate（locate）コマンドのデータベースを更新するシェルスクリプトで、cronデーモンによって毎朝実行されている。slocateコマンドは、ファイルやディレクトリのありかを教えてくれる便利なコマンドだ。たとえば、「Mesa」という文字列を含むファイルやディレクトリを探しているなら、次のようにすればよい。

```
$ slocate Mesa
```

```
/usr/X11R6/lib/libOSMesa.so
/usr/X11R6/lib/libOSMesa.so.3
/usr/X11R6/lib/libOSMesa.so.3.3
/usr/lib/libMesa.so.3.0
```

あっという間に4つのファイルが見つかった。findコマンドで全ディレクトリを検索するのに比べ、非常に高速だ。これは、毎朝更新されるデータベース（slocate.db）のおかげなのだ。

あなたの管理するLinuxマシンがネットニュースサーバだったり、Squidを動かしているサーバなら、膨大な数の記事ファイルやキャッシュデータがディスクに保存されているだろう。これらのファイルをslocateデータベースに加える必要はない。そこで、/etc/cron.daily/slocate.cronを書き換えよう。このファイルを開くと、/usr/bin/slocateコマンドを実行していることがわかる。slocateコマンドは-eオプションで検索除外ディレクトリを指定できる（すでに/tmpなどが書かれている）ので、ここにニューススプールやキャッシュディレクトリを書き加えれば作業は終了だ。明日から少しだけ静かな朝を迎えられるようになるだろう。

LILOで超える1024シリンダのカベ

WindowsとLinuxのデュアルブート環境を作る場合などに問題となるのがLILOの1024シリンダ問題だ（8Gバイト問題ともいわれる）。少し前までのLILOは、ハードディスクの1024シリンダ以降に置かれたカーネルにアクセスできず、Linuxを起動しようとすると画面に「LI」とだけ表示してフリーズしてしまった。

しかし、最近のLILOには、「lba32」というオプションが設けられており、1024シリンダ制限がなくなっているため、Linuxのインストール後にLILOを新しいものに入れ換えれば問題はすべて解決する。TurboLinux 6.0以降では、新しいLILOを採用しているので、インストーラで「lba32」オプションを選ぶだけでよい。

この方法を用いるためには、Linuxのインストール時に「ブートディスク」を作る必要がある。最初の起動にはこのディスクを使うのだ。ブートディスクにはカーネルイメージが書き込まれており、このイメージを使ってLinuxが起動する。/（ルート）パーティションなどは、Linuxをインストールしたハードディスクのものが使われる。

Linuxが起動したら、http://www.ibiblio.org/pub/Linux/system/boot/lilo/から最新版のLILOを入手し、展開して、make、make installする。次に、/etc/lilo.confをエディタで開き、先頭に「lba32」という行を追加する「linear」オプションが指定されていればこれを「lba32」に書き換える。最後に/sbin/liloを実行すれば作業は終了だ。

なお、Red Hat Linux 7.0Jでは、「lba32」オプションに対応したLILOがインストールされるが、インストーラからこのオプションを指定することができないので、ブートフロッピーで起動して/etc/lilo.confにオプションを追加しよう（/sbin/liloの実行を忘れないように）。

また、Kondara MNU/Linux 2000などでは、dev86パッケージに含まれるアセンブラのas86が古いため、LILOのmakeができない。編集部では、FTPサイトからdev86-0.15.3-1k.i586.rpmを入手してインストールすることでLILOをビルドした。

ログ管理を極める

定期的にログファイルの内容をチェックして、自分の管理するLinuxマシンがどのような状態なのか把握しておきたい。多くのディストリビューションでは、/var/logディレクトリにさまざまなログが出力される（**画面3**）。主要なログについて解説しよう。

●boot.log

Linuxの起動／シャットダウン時に出力されるログ。

●dmesg

Linuxの起動時に表示されるメッセージを記録したログ。dmesgコマンドで表示されるものとほぼ同じ。boot.logとあわせて内容をチェックすることで、システムの起動／終了時に起きている不具合を発見したり、その原因を探るのに使う。

●lastlog

各ユーザーが最後にログインした日時を記録している。内容を見るときは、lessなどのページャではなく、lastlogコマンドを使う。daemonなどログインを許可していない特殊なユーザーアカウントが使われていたら、何者かが不正に侵入した可能性が高い。

●maillog

sendmailによるメールの送信履歴を記録している。

●messages

さまざまなログが記録される。デーモンの起動に失敗しているときなど、問題が起きているときはまずこのログをチェックしよう。

●secure

認証関連の履歴を記録している。認証失敗の記録がたくさん見つかったら、不正な侵入を意図したアタックを受けている可能性がある。

●wtmp

そのマシンへのログイン履歴を記録している。内容を見るためにはlastコマンドを使う。

●xferlog

FTPサーバへのアクセス履歴を記録している。

これらのログは、klogdやsyslogdというデーモンによって収集され、ファイルに記録されている。

リアルタイムにログを調べたいときには、tailコマンドを使うと便利だ。

```
# tail -f /var/log/messages
```

このようにすれば、messagesファイルに新たに書き加えられる情報を常に見ることができる。

ログファイルには随時情報が追加されるので、だんだん大きくなって、ついにはディスクをパンクさせてしまう。そこで、ほとんどのディストリビューションにはログを分割し古いものからファイルを消去するlogrotateというプログラムが採用されている。

画面3のファイルで末尾に.1が付いているのが古くなってローテートされたファイルだ。logrotateはcronデーモンによって毎日1回起動され、/etc/logrotate.conf、/etc/logrotate.dディレクトリ以下の設定ファイルの内容に従ってログファイルをローテートする。

設定ファイルの書式はman logrotateで参照できるので、環境に合わせてカスタマイズするとよいだろう。

```
# ls -p /var/log/
XFree86.0.log  cron.1          maillog        pacct     sendmail.st  wtmp.1
XFree86.8.log  dmesg           maillog.1      ppxp/     spooler      xferlog
boot.log       fax/            messages       samba/    spooler.1    xferlog.1
boot.log.1     htmlaccess.log  messages.1     savacct   usracct
canna/         httpd/          netconf.log    secure    uucp/
cron           lastlog         netconf.log.1  secure.1  wtmp
```

画面3　ログディレクトリの内容

ユーザーを閉め出す！ ASCIIコードで文字入力！

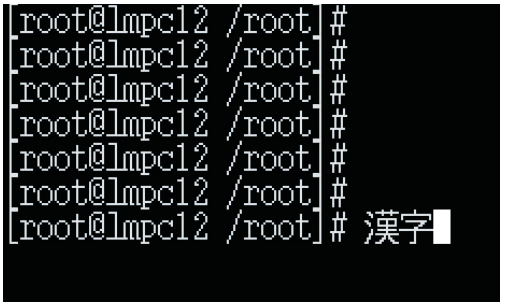
システムのメンテナンスを行う時など、一般のユーザーにログインしてほしくない場合はどうするか？　シングルユーザーモードで作業するのが常套手段だが、そこまでしなくてもよい場合は、/etc/nologinファイルを作成しよう。このファイルが存在すると、一般ユーザーはログインできなくなる。ログインしようとすると/etc/nologinファイルの内容を表示し、「Login incorrect」になるので、このファイルにはログインできない理由を書いておくとよい。

あまり無節操に使うとユーザーからの信頼を失うので、メンテナンスを行うときは事前に十分な告知をしておくことが重要だ。また、すでにログインしているユーザーに対しての効果はないので、現在の作業が終わりしだい速やかにログアウトするようお願いしよう。

さて、小技をもうひとつ。キーボードの設定が違っていて、どうしても入力できない文字があるときはどうするか？　コンソールでは、Altキーを押しながら、テンキーでASCIIコード（10進数で）を入力するとその文字を入力できる。たとえば、「｜」なら、Altキーを押したままテンキーから124と打ち、Altキーを離せば入力できる。いざというときには使えるかもしれない。

これを応用すれば、漢字だって入力できるのだ。konを起動して漢字が表示できるようにしておき、EUCコードを1バイトずつ入力する。たとえば、Alt＋180、Alt＋193で「漢」が入力でき、続いてAlt＋187、Alt＋250と打てば「字」が入力できる（**画面4**）。といっても、実用性は極めて低い。友達をビックリさせるのがせいぜいだろう。

画面4　かな漢字変換をせずに漢字を入力！（役には立たない）



manやhtmlをテキストにする

最近では、HTMLで書かれたオンラインドキュメントが増えてきた。これをプレーンテキストにしたいときはどうするか？　Netscape Communicatorを起動してコピー＆ペーストという方法もあるが、大量に文書がある場合は非常に面倒だ。こんなときは、テキストベースのブラウザを活用しよう。たいていのディストリビューションでは、Lynxというテキストベースブラウザがインストールされるのでこれを使い、

```
$ lynx -dump hoge.html > hoge.txt
```

とすればテキストに変換できる。テーブルやフレームに対応したw3m（http://ei5nazha.yz.yamagata-u.ac.jp/~aito/w3m/）というページャ／テキストベースブラウザを使っても同様のことができる。

```
$ w3m -dump hoge.html > hoge.txt
```

作成するテキストをWindowsや携帯端末など、シフトJIS環境のマシンで利用する場合は、nkfコマンドで漢字コードも変換しておくとよい。

```
$ w3m -dump hoge.html | nkf -s > hoge.txt
```

w3mは、ブラウザとしても非常に強力なので、ぜひインストールしておきたいソフトウェアだ。

では、manをテキストにするにはどうすればいいだろうか。manコマンドの出力をファイルにリダイレクトするとエスケープシーケンスが残るため、エスケープシーケンスによる画面制御を行わないプログラムでは非常に読みづらくなる。

エスケープシーケンスを取り除くには、colコマンドに-bオプションを指定して使うと便利だ。

```
$ man ls | col -b > ls.txt
```

知っておきたいデスクトップ
あれ？　inetd.confがない！？
デスクトップカスタマイズ（GNOME）

Red Hat Linux 7J（以下RH7J）でネットワークサーバを構築しようとして/etc/inetd.confを探した方も多いのではないだろうか。RH7Jでは、inetd（TCP wrappers）に代わり、xinetdというデーモンが採用されている。そのためinetdの設定ファイルは存在しないのだ。

外部からの接続に応じて各種ネットワークサーバを起動するという役割はinetd、xinetdともに同じだ。xinetdは、inetdに比べDoS（サービス不能）攻撃への対策がなされているなど、すっかり物騒になってしまった現在のインターネット環境を考慮して作られている。

■xinetdの設定ファイル

xinetdの設定ファイルは/etc/xinetd.confだ。この中には、サービスごとに{と}でくくったセクションを作り、その中で個別の設定を行う。RH7Jでは**リスト2**のような内容になっている。これを元に、設定内容を見てみよう。

defaultsセクションでは、すべてのサービスに有効な設定を行う。この例では、次のように設定されている。同時接続数は60、ログはsyslogd向けにauthprivファシリティで出力する。サーバへの接続が成功したら、クライアントのIPアドレス（HOST）とサーバのプロセスID（PID）をログに記録し、失敗したらIPアドレスとサーバが起動できなかった理由（RECORD）を記録する。

この後に、サービスごとのセクションを作ってもよいのだが、RH7Jではincludedirで/etc/xinetd.dディレクトリを指定して、このディレクトリに各サービスの設定ファイルを分けている。そこで、ここではFTPサーバ「wu-ftpd」の設定ファイル（**リスト3**）を例に内容を見ることにする。行番号は説明のために付けたものだ。

▼リスト2　Red Hat Linux 7Jの/etc/xinetd.conf

```
defaults
{
        instances               = 60
        log_type                = SYSLOG authpriv
        log_on_success          = HOST PID
        log_on_failure          = HOST RECORD
}

includedir /etc/xinetd.d
```

1行目は、このセクションで設定するサービス名を/etc/serviceに書かれている名前で規定している。3行目ではこのサービスが有効であることを設定している。disable=yesにするとこのサービスは利用できなくなる。4行目はソケットタイプの指定だが、これはサービスによって異なる。5行目は、このサービスが複数の接続を受け入れることを意味する設定だ。wait=yesにすると、1つの接続が終了するまで次の接続要求を受け付けない。6行目では、サーバを実行するユーザーIDを指定している。7行目がサーバの実行ファイルのパスで、8行目は引数として渡すオプションの指定だ。9、10行目はログに記録する内容だが、「+=」はデフォルト設定に加えて次の内容も記録するという意味になる。DURATIONはサーバへの接続時間、USERIDはidentプロトコルによって得られたユーザーの情報である。11行目はサーバの実行優先度の指定だ。数字が大きいほど優先順位は低くなる。

■そのほかの設定

ここまでで紹介したほかに、接続元をIPアドレスやホスト／ネットワーク名で限定するonly_from／no_access、接続を時刻で限定するaccess_timesなどが利用できる。これらはdefaultsセクションに設定してすべてのサービスに適用することもできるし、サービスごとに設定することも可能だ。

さらに、xinetdにはTCP接続をほかのマシンにリダイレクトする機能もある。この機能を使うと、外部ネットワークから直接接続できないマシンでネットワークサービスを行える。

このように、xinetdは強力かつ柔軟な設定ができるソフトウェアだ。今後、採用するディストリビューションも増えてくるだろう。なお、xinetd.confの詳しい書式は、man xinetd.confで調べられるの参考にしてほしい。

▼リスト3　Red Hat Linux 7Jの/etc/xinetd.d/wu-ftpd

```
 1: service ftp
 2: {
 3:         disable = no
 4:         socket_type       = stream
 5:         wait              = no
 6:         user              = root
 7:         server            = /usr/sbin/in.ftpd
 8:         server_args       = -l -a
 9:         log_on_success    += DURATION USERID
10:         log_on_failure    += USERID
11:         nice              = 10
12: }
```

bash、vi、Emacsの基本技

シェル（bash）は、Linuxをコマンドラインで使うときに常に利用する環境だ。また、/etcディレクトリ以下の設定ファイルやシェルスクリプトを変更したり、Cプログラムのソースを書き直したりするときは、やはりエディタにはviを使いたいという人も多い。一方、ログインからログアウトまで、すべてのことをEmacs内で実行する人もいるぐらいに完成された環境を備えたEmacsエディタも定番だ。これらは非常に奥が深いソフトウェアだが、使いこなしてこそ真のLinuxerといえよう。まずは基本から覚えてみよう。

便利なコマンド入力方法を覚えよう

コンソールやX Window Systemのターミナル（xtermやkterm、rxvtなど）から、コマンドを入力するときに利用するシェルプログラム（bash）は、キー入力を楽にするための機能をたくさん備えている。bashを、もっと便利に使おうではないか。

■ファイル名の補完

長いファイル名を入力するときに、全部間違えずに入力するのは大変だ。そういうときには、bashの補完機能を使おう。ファイル名の先頭の何文字かを入力して、Tabキーを押すとファイル名が補完される。

入力した文字から始まるファイルが複数存在する場合には補完されないので、もう一度Tabキーを押すと、該当するファイルがすべて表示される。その場合には画面を見て確定できるように文字を足してからTabを押せばよい。

なお、ファイルが多すぎる場合には、「Display all 1599 possibilities? (y or n)」のように注意されるので、nを押してから、絞り込めるように文字を足そう。

ディレクトリ名をたどっていく場合やコマンドを入力する際にも使える。

■ヒストリ

bashでは、実行したコマンドをファイル（~/.history）に記録している。上下の矢印キーを押すことで、そのヒストリをひとつずつ前後にたどることができる。すでに実行したコマンドを再度実行する場合には、入力し直して間違えることがないため便利だろう。

過去のヒストリを表示するには、historyコマンドを使う。ログインし直したり、再起動しても、ファイルに記録されていることも覚えておこう。

■コマンド行の編集

左右のカーソル移動キーを使ってカーソルを移動し、入力している文字を修正することは、すでにやったことがあるだろう。その編集作業をより便利にするために、いろいろなキー操作が行えるようになっている（**表1**）。表中の「Ctrl＋英字」は、Ctrlと英字キーを同時に押す。また、アンドゥ以下の4つは、

同時ではなく2回に分けて入力する。

■！を利用しよう

ディレクトリを作成して、そこにコピーしたりというように、直前のコマンドの引数を利用して、次のコマンドを実行することはよくあることだ。

この直前のコマンドの引数を、！＊（引数全部）、！$（最後の引数）として代用することができる。たとえば、

```
$ mkdir test
$ cp sample.c test
$ ls test
```

と入力する場合には、最後の引数を、！$を使って以下のように入力しても同じ結果となる。

```
$ mkdir test
$ cp sample.c !$
$ ls !$
```

また、！のあとにコマンド名を書くだけで、ヒストリから一致するコマンドを検索し、直近に実行した内容をそのまま実行してくれる。システム設定ファイルを修正し、サービスを再起動するときなど、いくつかの作業を繰り返して実行するときに便利だ。

動作	キー
入力ファイル名を補完	Tab
カーソルを行の先頭に移動	Ctrl＋A
カーソルを行の最後に移動	Ctrl＋E
←（カーソルを左に移動）	Ctrl＋Bまたは←
→（カーソルを右に移動）	Ctrl＋Fまたは→
前回実施したコマンド（ヒストリから）	Ctrl＋Pまたは↑
ヒストリにある次のコマンド	Ctrl＋Nまたは↓
最後からヒストリをサーチ	Ctrl＋R
カーソル位置から行末まで削除	Ctrl＋K
カーソル位置から行頭まで削除	Ctrl＋U
カーソル位置から1ワード削除	Alt＋D
カーソル位置から左へ空白までを削除	Ctrl＋W
削除した内容をペースト	Ctrl＋Y
アンドゥ	Ctrl＋X Ctrl＋U
キーボードマクロ開始	Ctrl＋X (
キーボードマクロ終了	Ctrl＋X)
キーボードマクロ実行	Ctrl＋X E

表1　bashのキー操作

たとえば、Apacheの設定ファイルを変更するときには、

```
# vi /etc/httpd/conf/httpd.conf
# /etc/rc.d/init.d/httpd restart
```

のようにするが、設定を間違えていることがわかったりしたら、

```
# !vi
# !/etc
```

とするだけで、さきほどのコマンドが実行されるのだ。

■ワイルドカード

複数のファイル名を指定するときには、？（1文字にマッチ）、＊（複数文字にマッチ）が使えるのは常識だが、大カッコ（ブラケット）を使うと特定の文字を表せて便利だ。

たとえば、[A-C] ＊のようにすると、AからCで始まるファイル名を表す。[^tsy] のように、^を付けると否定の意味になり、tとsとy以外の文字を表す。なお、ディレクトリ名にワイルドカードを使うこともできる（**画面1**）。

■alias（エイリアス）

よく使うコマンドは、aliasを設定しておくとキー入力を減らすことができる。

```
$ alias pl='ps aux | less'
```

のようにしておくと、plだけで、「ps aux ｜ less」を実行する。aliasを取り消すには、unaliasを使う。実際にはaliasの指定を、.bash_profileに書いておき、ログインしたときにbashに設定されるようにしておく。

```
# cd /usr/src/linux
# grep "Red Hat" */*/*.c
drivers/net/3c527.c: *  (c) Copyright 1998 Red Hat
Software Inc
drivers/sound/skeleton.c: *     (c) 1998 Red Hat
Software
drivers/sound/uart6850.c: * Extended by Alan Cox for
Red Hat Software. Now a loadable MIDI driver.
:
```

画面1　ディレクトリ名にワイルドカードを使った例

viエディタを気軽に使おう

すべてのLinuxディストリビューションに標準で含まれているviエディタは、Linux（UNIX）を利用するのならば最低限の使い方は覚えておきたいコマンドだ。ネットワーク環境が整っていなくてスタンドアローンで使うときなど、設定ファイルの書き換えが必要な場合に使用するからだ。

ここでは、viの基本的な操作方法を解説しよう。

■これだけは覚えておきたいキー操作

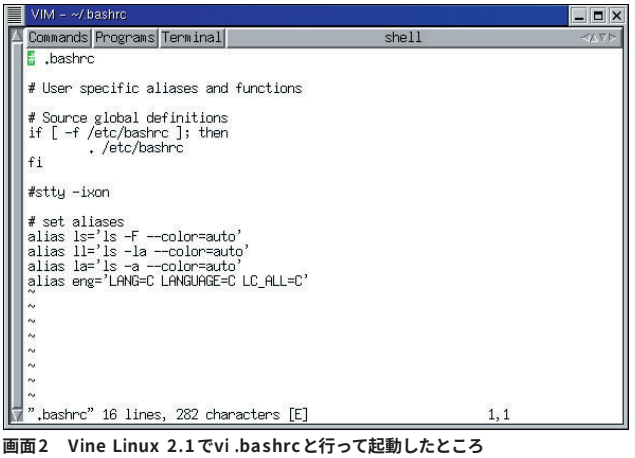
まず、viを起動するには「vi　＜ファイル名＞」のように編集したいファイル名を引数にして実行する。たとえば、.bashrcというファイルを編集するのならば、

```
$ vi .bashrc
```

とすると、画面上にファイルが読み込まれて表示される（**画面1**）。ファイルの終わりから下の部分には、“~”記号が表示されている。

viでは、コマンドモードと入力モードがあって、起動直後はコマンドモードになっている。たいていのWindows用のワープロやエディタはモードレスなので、いきなりキーを押して入力すればよいのだが、viでは入力モードにしてから文字の入力を行うのが慣れるまでは面倒な点だ。

なお、一番下の行（通常は24行目）は、ステータスが表示されたり、：や/などで始まるコマンドを入力するときの表示エリアとなっている。



画面2　Vine Linux 2.1でvi .bashrcと行って起動したところ

■カーソル移動

カーソルを1文字分移動するためには、コマンドモードで、h（左）、j（下）、k（上）、l（右）の4つのキーを利用する。矢印キーも利用可能なのでそれを使用してもいいが、わざわざ右手のホームポジション上にあるキーが割り当てられているように、ブラインドタッチ入力が行える人ならば、hjklキーでカーソル移動するのが通というものだ。

なお、カーソルは文字のない場所には移動できない。また、行末でl（右）キーを押したり、行の先頭でh（左）キーを押しても、カーソルはそのままで次の行には移動しない。そして1行が80文字を超えるような長い文を編集する場合には、画面上の1行単位ではなく、論理的な行単位でカーソルが移動することに注意してほしい。

■文字の入力

文字を挿入するには、iまたはaキーを押す。このキーを入力すると最下行に「INSERT」と表示され、入力モードになる。iを押してもカーソルは動かないが、次に入力した文字列は、カーソルがある文字の前に挿入されていく。aならば、カーソルのあった文字の右側から挿入される。

入力が終了したら、ESCキーを押してコマンドモードに復帰する。

■文字（行）の削除

カーソル位置の文字を1文字削除するには、xキーを押す。複数の文字を削除する場合には、10x（10文字削除）のように文字数を入力してからxを押すことで可能だ。

カーソル行自体を削除するには、ddと入力する（10ddなら

動作	キー
↑（カーソルを上に移動）	k
↓（カーソルを下に移動）	j
←（カーソルを左に移動）	h
→（カーソルを右に移動）	l（小文字のL）
カーソル位置の前に文字挿入（入力モードへ）	i
カーソル位置のあとに文字挿入（入力モードへ）	a
入力モードからコマンドモードへ	ESC
カーソル位置の1文字削除	x
カーソル位置の行削除	dd
ファイルに保存して終了	ZZ
保存せずに強制終了	：q！＜Enter＞

表2　viの必須キー操作

10行削除)。このようにコマンドの前に数値を入力することで繰り返し実行する方法は、多くのほかのコマンドでも有効なので試してみるとよいだろう。

カーソル位置から行末までを削除したいときは、Dとする。

■viの終了のしかた

修正した内容を保存して終了するのならば、ZZと入力する。保存しないで終了するのならば、：q！と押して、最下行に「：q！」と表示されたところでEnterキーを押せばよい。

■viをもっと便利に使うには

ここまでの使い方を覚えておけば、とりあえずファイルを修正することは可能だろう。次にもっと便利に使うための操作を紹介しよう。

■高速なカーソル移動法

まず表示されている画面内での移動方法だが、単にEnterキーを押すと次の行の先頭にカーソルが移動する。また、行の先頭にカーソルを移動するには0（ゼロ）を、行末に移動するには$を押す。そして、表示されている画面左上に移動するにはHを、画面中央はM、画面左下にはLを押せばよい。

大きなファイルのときには、1文字単位でカーソルを移動するのは不便だ。Ctrl＋B（アップ)、Ctrl＋F（ダウン）で1画面単位でスクロールする。画面半分ずつなら、Ctrl＋U（アップ)、Ctrl＋D（ダウン）だ。こちらのほうが覚えやすいかも(Up＝U、Down＝D)。ファイルの最後にジャンプするにはGを、ファイルの先頭へは1Gと入力すればよい。

また、何行目かがわかっているのならば、：に続けて行番号を入力し、Enterを押せば直接その行にジャンプする。行番号を表示するには、：set numberとする（やめるには：set nonumber)。

■アンドゥ（やり直し）

操作を誤った場合には、uキーを押して直前に実行したコマンドをキャンセルできる。

■コマンドの繰り返し

最後に行ったコマンドを繰り返して実行するには、.（ピリオド）を押す。たとえば、ディレクトリ名やURLなどの長い名前を複数の場所に挿入する場合に便利だろう。

■行の連結

viでは、行頭や行末で文字の削除を行っても、次の行をつなげることはできない。カーソル行に次の行を連結するコマンド、Jを利用しよう。カーソル位置は行内のどこであっても構わない。このとき、次の行の先頭部分の空白は削除され、連結部分に1文字分の空白が挿入されるということに注意しよう。

■文字列の検索

文字列を検索するには、/に続けて検索したい文字を入力し、Enterを押す。カーソル位置から検索し、最初に見つかった場所にカーソルが移動する。同じ文字列を続けて検索するのならば、/とEnterだけでよい。ファイルの先頭（前方）に向かって検索する場合には、/の代わりに？を使う。

なお、検索中にファイルの最後（？ならば先頭）まで行くと、自動的にファイルの先頭（最後）から継続して検索される。

■コピー＆ペースト

文字列のコピー＆ペーストを行うには、まずコピーしたいところにカーソルを移動し、yyでバッファにコピーする。数値を入力してからyyを押せば、その行数分がバッファに入る。

そしてペーストしたい場所にカーソルを移動し、P（カーソル行の前へ挿入）または、p（カーソルの次の行へ挿入）キーを押す。カット＆ペーストならば、yyの代わりにddで削除すればよい。

動作	キー
画面の左上にカーソル移動	H
画面の最下行にカーソル移動	L
行の先頭にカーソル移動	0（数字のゼロ）
行の末尾にカーソル移動	$
1画面分スクロールアップ	Ctrl＋B
1画面分スクロールダウン	Ctrl＋F
N行目にジャンプ	：<N><Enter>
ファイル（文章）の先頭にジャンプ	1G
ファイル（文章）の最後にジャンプ	G
カーソル行の前に入力（入力モードへ）	O
カーソル行のあとに入力（入力モードへ）	o
アンドゥ（やり直し）	u
コマンドの繰り返し	.
カーソル行と次の行をつなげる	J
文字列検索（後方検索）	/文字列<Enter>
文字列検索（前方検索）	?文字列<Enter>
カーソル行をバッファにコピー	yy
バッファの内容をカーソル行の前にペースト	P
バッファの内容をカーソル行のあとにペースト	p

表3　viの便利なキー操作

Emacsで快適環境を実現しよう

Emacsは、単なるエディタとしてテキストを編集するだけではなく、MewやWanderlustといったメーラや他のプログラムと連係することで、より便利に使うことができる。

コンソールからも利用できるが、X Window Systemで実行すれば、メニューが表示され、マウスとカーソルキーでほとんどの操作が行えるため、Windowsアプリケーションとほぼ同様に扱える（**画面3**）。そうはいってもEmacsの基本は押さえておく必要がある。

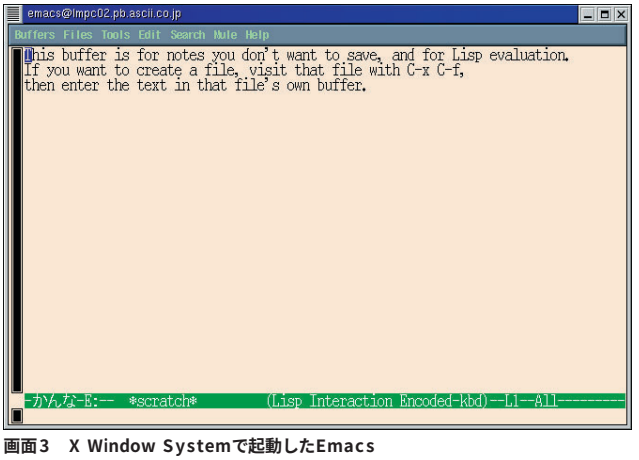
■これだけは覚えておきたいキー操作

まず、Emacsを起動するには、「emacs　＜ファイル名＞」のように編集したいファイル名を引数にして実行する（単にemacsだけでも構わない）。

Emacsのキー操作は、Ctrlキーとメタキー（通常はAltに割り当てられている）を多用する。CtrlとBを同時に押すような場合には、Emacs風では「C-b」のように記述するが、今回はbash／viと合わせるために「Ctrl＋B」と表記している。メタキーの場合も同様に「M-v」を「Alt＋V」と表記した。

WindowsからtelnetでEmacsが使えない場合には、メタキーの代わりに、ESCキーを押してからコマンドのキーを押すという2ストロークで入力できる。

一連のコマンドを入力中にキーを押し間違えてしまったときには、Ctrl＋Gを押すことでキャンセルできる（アンドゥではないことに注意）。



画面3　X Window Systemで起動したEmacs

Emacsには、viのようなモードはないので、いきなりキーを叩いていけば、文字を入力することができる。なお、一番下の行（通常は24行目）は、コマンドを入力するときの表示エリアで、その上の反転している行に入力モードが表示される。

■カーソル移動

カーソルを1文字分移動するためには、Ctrl＋B（左）、Ctrl＋F（右）、Ctrl＋P（上）、Ctrl＋N（下）を使う。それぞれ、Backward／Forward／Previous／Nextの頭文字である。矢印キーも利用可能だ。

■文字の削除

カーソル位置の文字を1文字削除するには、Ctrl＋Dを押す。Deleteキーならばカーソルの左の文字が削除される。複数の文字を削除する場合には、Ctrl＋U、10、Ctrl＋D（10文字削除）のように文字数を指定することもできる。このように繰り返し回数を入力する場合には、Ctrl＋Uに続けて数値を指定する。この方法は、多くのほかのコマンドでも有効なので試してみるとよいだろう。

なお、Backspaceキーは通常ヘルプ機能に割り当てられている。不便な場合には、BackspaceをDeleteキーの代わりに割り当てるように変更しよう。.emacsファイル（または.emacs.el）に、下記のように追加しておけばよい。

```
(load-library "term/bobcat")
```

カーソル位置から行末までを削除するのならば、Ctrl＋Kを押す。

■ファイルをオープンする

新しくファイルを読み込むには、Ctrl＋X、Ctrl＋Fと入力し、最下行に「Find file: ~/」と表示されるのでファイル名を入力する。このときに、bashと同じようにTabキーでファイル名の補完が利く。このときに複数ファイルがある場合は、一時的にウィンドウが開いてディレクトリの内容を表示してくれる。

■Emacsの終了のしかた

修正した内容を保存するには、Ctrl＋X、Ctrl＋Sと入力する。ファイル名を指定して保存する場合には、Ctrl＋X、Ctrl＋W、ファイル名だ。

終了するのならば、Ctrl＋X、Ctrl＋Cと押す。もしファイルが保存されていない場合には、最下行に保存するか確認のメッセージが表示されるので、yまたはnで答え、n（保存しない）ならば、「Modified buffers exist; exit anyway? （yes or no）」ともう一度確認される。ここでyesを入力すると修正した内容は破棄される。

■高速なカーソル移動法

行の先頭にカーソルを移動するには、Ctrl＋Aを、行末に移動するにはCtrl＋Eを押す。1画面分のスクロールは、Alt＋V（アップ）、Ctrl＋V（ダウン）で行う。ファイルの先頭にジャンプするにはAlt＋<を、ファイルの最後へはAlt＋>を入力すればよい。

■アンドゥ（やり直し）

操作を誤ったときには、Ctrl＋_キー（Ctrl＋X、uの場合もある）を押して直前に実行したコマンドをキャンセルできる。

■文字列の検索

文字列を検索するには、Ctrl＋Sに続けて検索したい文字を入力していくと、その時点で一致する文字列を検索し、画面上でカーソルが移動する。見つかったならばカーソル移動キーを入力して、検索コマンドから抜ける。

同じ文字列を検索するのならば、続けてCtrl＋Sだけでよい。ファイルの先頭（前方）に向かって検索する場合には、Ctrl＋Rを使う。なお、検索中にファイルの終端まで行くと、ベルが鳴り警告されるが、もう一度キーを押せばファイルの先頭（最後）から継続して検索する。

■コピー＆ペースト

文字列のコピー＆ペーストを行うには、まずコピーしたい範囲を指定するためにカーソルを移動し、Ctrl＋＜SPC＞（スペースキー）でマークをセットする。次にカーソルを移動し、Ctrl＋Wを押すとマークからカーソルまでの範囲がキルリングと呼ばれるバッファにコピーされる。

そしてペーストしたい場所にカーソルを移動し、Ctrl＋Yを押せばキルリングバッファの内容が張り付けられる。カット＆ペーストならば、Ctrl＋Wの代わりにAlt＋Wで削除すればよい。

マークしたい位置がわからなくなったら、Ctrl＋X、Ctrl＋Xとするとマークとカーソル位置が入れ替わるので確認できる。カーソル位置から行末までを削除してキルリングへコピーするのならば、マークは不要でCtrl＋Kと押すだけでよい。

動作	キー
↑（カーソルを上に移動）	Ctrl＋Pまたは↑
↓（カーソルを下に移動）	Ctrl＋Nまたは↓
←（カーソルを左に移動）	Ctrl＋Bまたは←
→（カーソルを右に移動）	Ctrl＋Fまたは→
カーソル位置の1文字削除	Ctrl＋D
カーソル位置の前の1文字削除	DEL
入力キーのキャンセル	Ctrl＋G
ファイルを開く	Ctrl＋X Ctrl＋F ファイル名
バッファをの内容をファイルに保存	Ctrl＋X Ctrl＋S
保存せずにバッファの内容を消去	Ctrl＋X Ctrl＋K
Emacsを終了（保存していない場合は確認される）	Ctrl＋X Ctrl＋C

表4　Emacsの必須キー操作

■かな漢字変換のOn／Off

Linux上のかな漢字変換ソフトをEmacsから使うには、Ctrl＋￥と押すことで、On／Offを切り替えることができる。

■ウィンドウ操作

Emacsでは、画面を分割して作業することが可能だ（そのひとつひとつをウィンドウと呼ぶ）。Ctrl＋X、2と押すと現在のウィンドウを2つに分けることができる。すぐ下のウィンドウに移動するには、Ctrl＋X、oと押す。そこでファイルをオープンすれば、同時に2つのファイルを編集できるのだ。

動作	キー
行の先頭にカーソル移動	Ctrl＋A
行の末尾にカーソル移動	Ctrl＋E
1画面分スクロールアップ	Alt＋V
1画面分スクロールダウン	Ctrl＋V
ファイル（文章）の先頭にジャンプ	Alt＋<
ファイル（文章）の最後にジャンプ	Alt＋>
コマンドの引数を設定する	Ctrl＋U 数値　コマンド
アンドゥ（やり直し）	Ctrl＋_またはCtrl＋X u
文字列検索（後方検索）	Ctrl＋S 文字列
文字列検索（前方検索）	Ctrl＋R 文字列
マークをセット	Ctrl＋@またはCtrl＋<SPC>
マークとカーソル位置を交換	Ctrl＋X Ctrl＋X
マークからカーソルまでの範囲を削除してキルリングへ	Alt＋W
マークからカーソルまでの範囲をキルリングへコピー	Ctrl＋W
カーソル位置から行末までを削除してキルリングへ	Ctrl＋K
キルリングの内容をペースト	Ctrl＋Y
かな漢字変換のOn/Off	Ctrl＋￥
ウィンドウを分割する	Ctrl＋X 2
すぐ下のウィンドウに移動	Ctrl＋X o
バッファをウィンドウに呼び出す	Ctrl＋X b バッファ名
バッファの一覧表示と操作	Ctrl＋X Ctrl＋B
表示ウィンドウを削除	Ctrl＋X 0（数字のゼロ）

表5　Emacsの便利なキー操作

連載特集

申し込みから独自ドメイン運用まで徹底解説！

フレッツ・ISDNで常時接続インターネットサーバ!

独自サーバで自分だけのインターネットサイトを構築しよう

前回は、フレッツ・ISDNとプロバイダの申し込みまでを解説した。今回はいよいよサーバを準備して、独自ドメインを取得して常時接続環境を実現する。いよいよ独自ドメイン運用の第一歩だ。

今回はかなり早足になる。独自ドメインを運用するには、いくつも同時に設定をしなければならないからだ。たとえば、常時接続であればセキュリティにも十分注意しなければならない。とはいっても何はともあれサーバがなければ始まらない。まずはサーバの設定からだ。

ACT. 2

ISDN フレッツ 常時接続
サーバを立ち上げよう！

フレッツ・ISDNが開通すると、地域IP網に接続するための専用電話番号（ダイヤルアップ番号）が通知される。申請した電話回線から、このダイヤルアップ番号へのデジタル接続のみがフレッツ・ISDNによる定額接続料の対象となる。違う電話回線や、プロバイダのアクセスポイントへの接続は対象外となるので注意してほしい。つまり、違う場所からダイヤルアップ番号に電話した場合や、直接プロバイダに接続した場合は別に課金されることになる。

プロバイダ登録の完了

プロバイダの登録が完了すると、ネットワークIDとネットワークパスワードが送付される。ダイヤルアップ番号に接続した際に必要となるのが、このネットワークIDと、ネットワークパスワードだ。といっても、イメージとしてはプロバイダのアクセスポイントに接続した際に必要となる、接続IDとパスワードと同じだ。唯一違うのは、複数のプロバイダと契約している場合、このネットワークIDとネットワークパスワードによって、接続するプロバイダを変更できる点だ。

なお、ネットワークIDは、「xx1234567@boo.hogehoge.ne.jp」といった、一見、メールアドレスのようなものだ（メールアドレスではないので注意）。

また、編集部では今回、固定（静的）グローバルIPアドレスを提供してくれる、NTT PCコミュニケーションズのBiz IP8に申し込んだので、8つの固定グローバルIPアドレスも一緒に割り当てられた。

さて、ダイヤルアップ番号と、ネットワークID、ネットワークパスワードが入手できたら、ネットワークの構築にとりかかろう。

ネットワーク計画

実際にサーバをセットアップする前に、ネットワークの計画を立てておいたほうがいいだろう。ネットワークの計画といっても大げさなものではなく、IPアドレスをどのように割り振るかということを考えておくだけだ。

今回のネットワークでは、まず最初にダイヤルアップルータと、サーバの2つを設置することにする。そのため、必要なIPアドレスはダイヤルアップルータとサーバの2つになる。プロバイダから割り当てられた固定グローバルIPアドレスは8つだ。このアドレスを割り当てることになるのだが、ここでIPアドレスの決まり事をおさらいしておかなければならいない。

今回編集部で申し込んだBiz IP8では、111.122.133.128/29というIPアドレスが割り当てられる（アドレス部の数値は架空）。29という数値は、ネットマスクを表している。IPアドレスには、ネットワーク部とホスト部が含まれており、どこまでがネットワーク部で、どこからがホスト部であるかを示すものがネットマスクだ。

これは2進数でないとわかりづらいのだが、/29とは、29ビット目までがネットワーク部で、残り3ビットがホスト部だということを示している。

3ビットで表せる数は、0（000）～7（111）となるので、8個のIPアドレスがホスト部に割り当てられているということになる。このことから、割り当てられた固定グローバルIPアドレスは、111.122.133.128～111.122.133.135までだということがわかるのである（**図1**）。この形式での表記は、ネットワークプレフィックスと呼ばれ、従来の255.255.255.0というようなネットマスクに代わる表記として使われている。/29を従来のネットマスクの表記で記述すると、255.255.255.248となる。

割り当てられたIPアドレスのうち、最初と最後のアドレスは機器に割り当

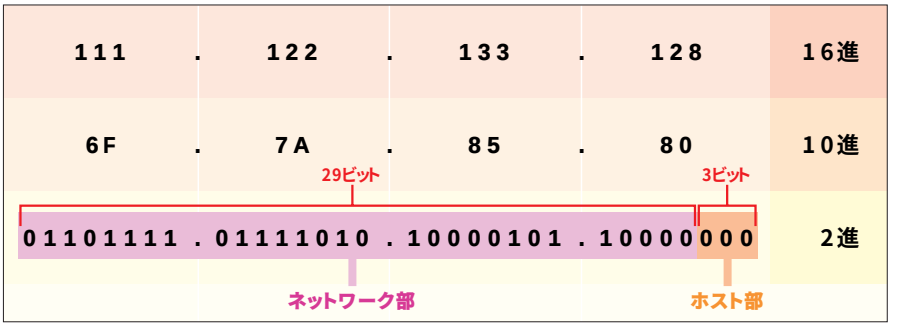


図1　ネットワークプレフィックスによるIPアドレス表記

てることはできない。先ほどのアドレスの例では、111.122.133.128と111.122.133.135がこれに相当する。

最初のアドレスは、ネットワークそのものを指すアドレスで、最後のアドレスはブロードキャスト用に使用されるものである。つまり、実際の機器に割り当てられるアドレスは、111.122.133.129～111.122.133.134までの6個ということになる。

実際のIPアドレスの割り振りだが、一般的に、最初に使用できる最も若いIPアドレスをルータに割り当てることが多いので（この例では111.122.133.129）、この慣例に従うことにする。サーバはその次のアドレスに割り当てることにしよう。

これによって、ネットワークの構成が次のようになる。

ダイヤルアップルータ

```
111.122.133.129
```

（デフォルトゲートウェイ）

サーバ

```
111.122.133.130
```

上記の例を参考に、実際に割り当てられた固定グローバルIPアドレスを使ってネットワーク計画を立てて欲しい（**図2**）。

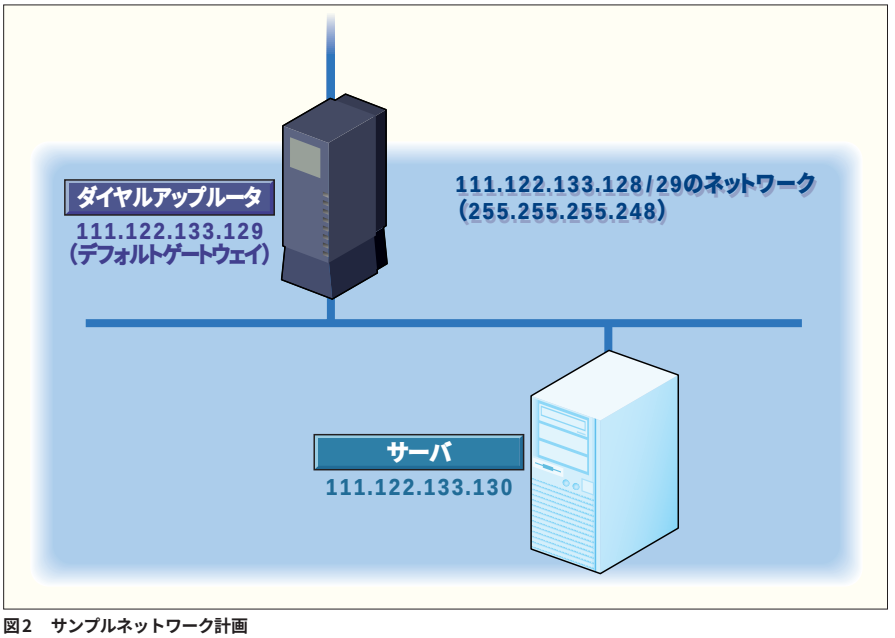


図2　サンプルネットワーク計画

サーバの準備

いよいよサーバを準備してインターネットに接続する。早速マシンの準備だ。

編集部では、先月号の『XFree86セットアップ徹底ガイド』でも使用した、ベアボーンキットをもう1台用意した（**写真1**）。マザーボードはGigaByteのGA-60MM7Eで、チップセットは統合型のi815Eを採用した最新のものだ（**写真2**）。CPUはCeleron 667MHzをセレクト。最新のマザーボードにしてはショボイCPUを選択したが、サーバ用途なので速度はそんなに重視する必要はないし（それでも速いCPUだが）、マザーボードが最新なのは、グラフィックチップ統合型の安いベアボーンキットを選択した結果だ。

ダイヤルアップルータについて

ダイヤルアップルータはサーバをインターネットに接続する際の必須アイテムだ。サーバはダイヤルアップルータを経由してインターネットに接続する。

今回は、編集部に転がっていたヤマ



写真1



写真2



写真3

ハのRTA50iを利用することにする（**写真3**）。若干古い機種だが、特に問題はないはずだ。ただし、古い機種ではフレッツ・ISDNに対応していない場合もある。フレッツ・ISDNといっても、通常のダイヤルアップで接続するのと同じなので、機種などは関係ないと思うかもしれない。しかし、多くのダイヤルアップルータは、なるべく回線を切断するように設計されているため、フレッツ・ISDNのような常時接続でも回線を切断してしまうことがある。これは、意図しない接続による大量課金が発生しないためのメーカーの配慮だが、フレッツ・ISDNに限っていえば、勝手に接続が切られてしまってはかなわない。そのために、フレッツ・ISDNに対応しているダイヤルアップルータが必要となるのだ。

なお、古い機種でもフレッツ・ISDNに対応した新しいファームウェアをメーカーが提供していれば、フレッツ・ISDNで利用することも可能だ。

ダイヤルアップルータの設定は、マシンがセットアップされていないとできないので、先にLinuxをインストールすることにしよう。

Linuxのインストール

今回はサーバ構築ということで、どのディストリビューションにするかを悩んだが、結局Red Hat Linux 7J（以下、Red Hat 7J）にすることにした。サーバ用ディストリビューションを選択してもよかったのだが、ホームサーバとしての運用も考えると、デスクトップ環境も兼ねたいだろうし、12月号の付録CD-ROMに収録してあったので読者が入手しやすいだろうという理由からだ。また、標準でi815に対応しているXFree86が収録されている点も選択した理由のひとつだ。

なお、Red Hat 7Jを使用する場合は、ハングアップするというバグがあるので、Red Hat 7Jの改訂版に含まれるアップデート必ず行うこと（本誌1月号に収録）。

Linuxのインストールについては特に触れる必要はないと思うが、インストール時の「インストールの種類」で「カスタムシステム」を選択し、不要なパッケージはインストールしないようにしよう（**画面1**）。不要なパッケージやサービスはセキュリティホールとなる可能性があるからだ。

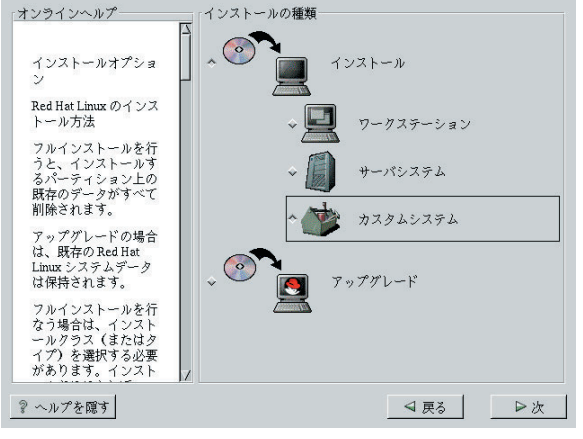
本来なら、X Window Systemもインストールしないほうがよい。とはいっても、ホームサーバなどでは、デスクトップ環境としても使いたい場合もあるだろうし、サーバ上でもGUIの設定ツールを使いたいこともあるだろうと思うので、各自必要なものを入れるといいだろう。X Window Systemはデフォルトで起動しないように、「グラフィカルログインの使用」をチェックしないようにするだけでもいいだろう。

実際に編集部のサーバにインストールした主要なパッケージは次のようなものだ。

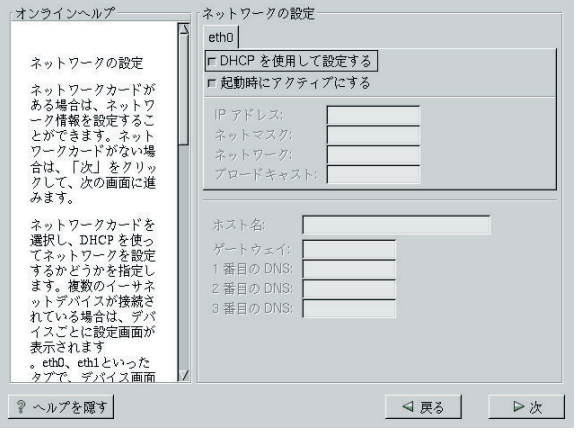
X Windowシステム
GNOME
メール／WWW／ニュースツール
ネットワーク接続ワークステーション
ニュースサーバ
SMB
Webサーバ
DNSサーバ
ユーティリティ

DNS（ネームサーバ）はドメインを取得する際に必須となるので忘れずにインストールしておこう。

また、「ネットワークの設定」では、とりあえず「DHCPを使用して設定する」を選択して、ダイヤルアップルータのDHCPを利用してIPアドレスを動的に割り当てるようにしておこう（**画面2**）。これは、ダイヤルアップルータを設定するまでの仮の設定だ。ダイヤルアップルータによっては、同じネットワーク内のマシンからでないと設定



画面1　「インストール種類」で「カスタムシステム」を選択



画面2　「ネットワークの設定」で、「DHCPを使用して設定する」を選択

ができないものがあるからだ。

インストール時の注意点は以上だ。Red Hat 7Jの詳しいインストール方法は、12月号の242ページを参照するといいだろう。

インストールが完了したら、Red Hat 7Jアップデートを当てることを忘れないようにしよう。

ダイヤルアップルータの設定

ダイヤルアップルータの設定方法は、使用する機種によって異なるので、マニュアルをよく参照してほしい。ここでは、編集部で使用するRTA50iを例に、なるべくほかのダイヤルアップルータにも当てはまるように説明する。

ダイヤルアップルータの多くは、Webブラウザや、telnetコマンドを使って設定を行う。今回使用した、RTA50iも同様だ。せっかくX Window Systemもインストールしたので、GUIで行いたいところだ。ということで、Netscapeを起動し、ルータのIPアドレスにアクセスする。ルータのIPアドレスはまだ指定していないので、工場出荷時のIPアドレスを指定することになる。多くの場合は、「192.168.0.1」というアドレスが設定されているだろう（**画面3**）。

多くのダイヤルアップルータでは、接続方法に「端末型接続（ダイヤルアップ接続）」と、「ネットワーク型接続」の2種類が選択できるようになっている。簡単にいえば、端末型接続は、通常のダイヤルアップ接続を行う場合の設定で、ネットワーク型接続はサーバを構築する場合に使用する接続だ。ここでは、「ネットワーク型接続」を選択する……といいたいところだが、RTA50iのネットワーク型接続では、ISDNによる接続が選択できなかった。どうやら、RTA50iは古い機種のためフレッツ・ISDNに対応していないようだ。早速、ヤマハのホームページで確認しよう……と思ったら、ルータの設定が終わっていないのでインターネットに接続できないではないか。

仕方がないので、「端末型接続」を選択して、通常のダイヤルアップのように接続することにする。フレッツ・ISDNといっても、通常のダイヤルアップと変わらない方法で接続することは可能だ。

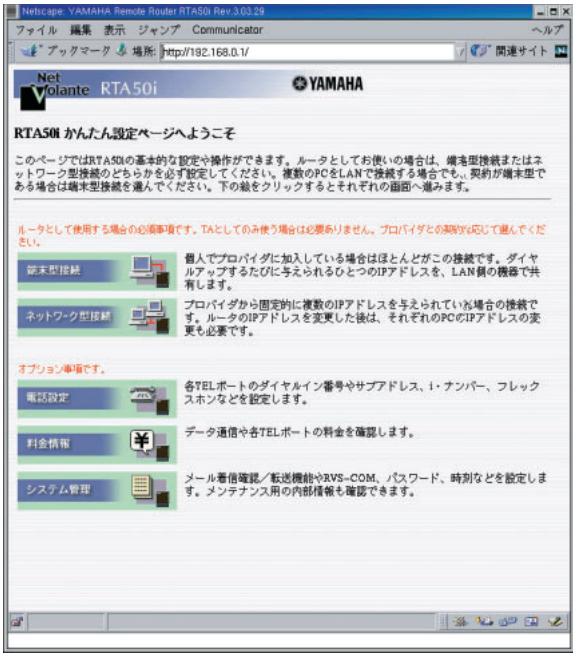
とりあえず接続してみることで開通試験もできるというものだ。

端末型接続を選択して、プロバイダの電話番号にNTTから指定されたダイヤルアップ番号を指定し、ユーザーIDにネットワークIDと、パスワードにネットワークパスワードを指定する。プライマリDNSとセカンダリDNSはプロバイダのネームサーバのアドレスを指定する（**画面4**）。

ここまで入力したら、［登録］ボタンをクリックしてプロバイダ情報を保存しよう。最後に自動接続設定を「する」にして［更新］ボタンをクリックする。

これで、インターネットへの接続要求があったときに、自動的に接続されることになる。Netscapeを起動してホームページが表示できるかどうか確認してみよう。

接続できない場合は、もう一度ダイヤルアップルータの設定を確認してみよう。また、ルータのISDN番号に契約した番号以外を指定している場合や、



画面3　RTA50iの設定ページ（192.168.0.1）



画面4　端末型接続による接続設定（RTA50i）

発信者番号通知サービスで「常に通知しない」として契約した場合は、ダイヤルアップ番号に「186（発信者番号通知）」を付ける必要がある。

無事ホームページが表示できれば、開通試験はOKだ。

インターネットに接続できたところで、ヤマハのホームページ（http://www.yamaha.co.jp/）を参照してみる。どうやら、古いバージョンのファームウェア（3.03.29）ではフレッツ・ISDNに対応していないらしく（当然だ！）、新しいファームウェア（3.05.35）にすればフレッツ・ISDN対応のメニューが表示されるらしい。早速ダウンロードしてアップデートすることにする。

ちょっと回り道をしてしまったが、これでネットワーク型接続が可能になるはずだ。ヤマハ以外のルータの場合でもおおむね同様だと思われるので、メーカーのホームページを確認してほしい。

さて、ファームウェアをアップデートしたら、ネットワーク型接続を設定しよう。まず、ダイヤルアップルータのIPアドレスを指定する。これは先ほどのネットワーク計画では、111.122.133.129だ。続いて、ネットマスクを指定する。「/29」という指定か、「255.255.255.248」という指定となる。

回線の種類は、「ISDN」を指定し、「フレッツ・ISDN」か、「常時接続」または、「IP接続サービス」といった項目を指定する。プロバイダの電話番号やユーザーのIDとパスワードには先ほどと同じく、NTTのダイヤルアップ番号、ネットワークIDとネットワークパスワードを指定する。

DNSに関しては、プロバイダのDNSを設定しておく。NATやIP masqueradeなどは、とりあえず使用しない設定にしておこう（**画面5**）。

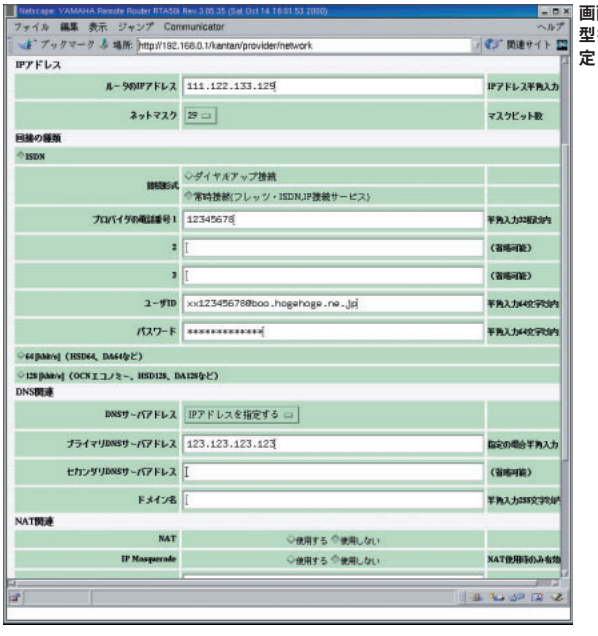
設定が終了したら、［登録］ボタンをクリックして登録する。なお、多くのルータは、設定を有効にするために一度電源を切る必要がある。

サーバの設定

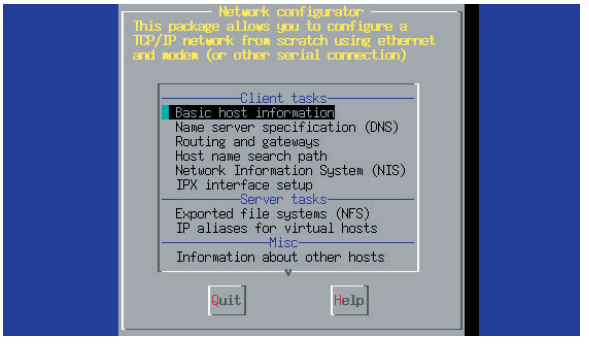
ルータの設定が終わったら、サーバのIPアドレスも固定グローバルIPアドレスに変更しておこう。インストール後にIPアドレスを変更するにはlinuxconfやnetconfコマンド（Turbo Linuxではturbonetcfgコマンド）を使用する。ここではnetconfコマンドを使用して変更する（**画面6**）。

まず、「Basic host information」を選択し、DHCPを使用しないように変更し、IPアドレスとネットマスクにプロバイダから与えられた値を指定する。サーバなので、ダイヤルアップルータにDHCP機能があったとしても、固定アドレスを指定するようにしよう。また、このIPアドレスは、インターネットに公開されるものなので、絶対に間違えないように気を付けてほしい。IPアドレスを間違えた場合はほかのサーバに迷惑がかかるのはもちろん、ネットマスクを間違えた場合は無用なパケットをインターネットに送信してしまうなど、ほかのネットワークに迷惑をかけてしまうからだ。くれぐれも気を付けるようにしたい。先ほどのサンプルネットワークでは、サーバのIPアドレスを111.122.133.130に、ネットマスクを255.255.255.248に設定することになる。

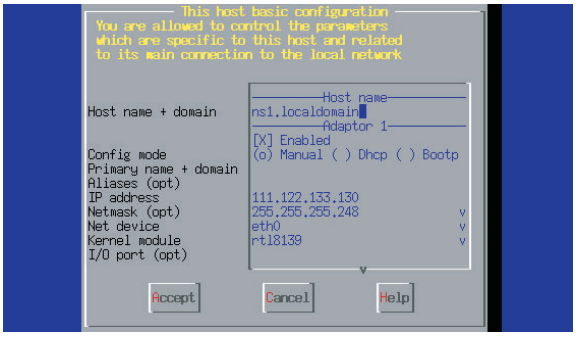
「Host name」はサーバのマシン名を仮の名前で設定しておこう。flsvr00や、将来ネームサーバになることを考えて、ns1などとしておくといいだろう。ドメイン名は仮なので、実際に存在しないような架空のものを指定しておけばいいだろう（**画面7**）。



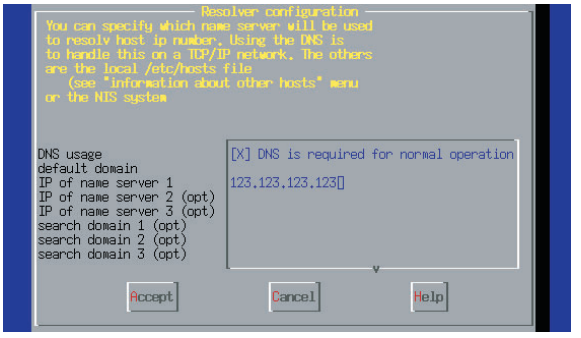
画面5　ネットワーク型接続による接続設定（RTA50i）



画面6　netconfコマンドによるネットワークの設定



画面7　「Basic host information」での設定



画面8
「Name server specification (DNS)」での設定

ここでの設定は以上だ。サンプルのネットワーク計画では、**表1**のような設定になるはずだ。

次に、「Name server specification (DNS)」を選択する。ここではDNSに関する設定を行うのだが、まだDNSサーバを立ち上げていないので、プロバイダのDNSを指定しておく。「IP of name server」にプロバイダのDNSを入力すればよい（**画面8**）。

最後に、「Routing and gateways」を選択し、デフォルトゲートウェイを設定する。「Set Defaults」を選んで、ダイヤルアップルータの固定グローバルIPアドレス（サンプルのネットワーク計画では、111.122.133.129）を指定し、「Enable routing」をチェックして有効にする（**画面9**）。

設定が終わったら、「Activate the changes」を選択し、サーバをリブートするか、次のようにしてネットワークを再起動する。

```
# /etc/rc.d/init.d/network restart
```

または、

```
# /etc/inid.d/network restart
```

Host name + domain	ns1.localdomain
Config mode	Manual
IP address	111.122.133.130
Netmask	255.255.255.248

表1　サンプルのネットワーク計画での設定値

これでサーバも固定グローバルIPアドレスに設定されたはずだ。

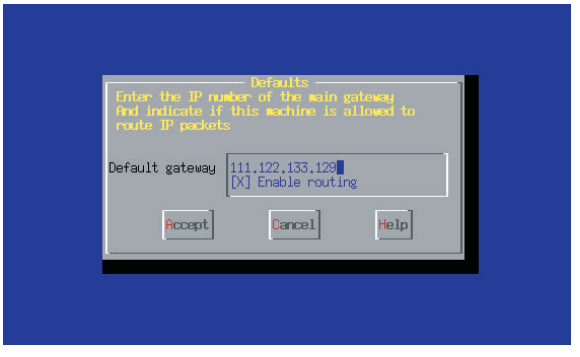
念のために設定値を確認しておこう。ifconfigコマンドを実行して、eth0に割り当てられたIPアドレスを確認してみる（**画面10**）。

ここでもう一度Netscapeを起動して、Webページが表示できるか確認する。ルータが正常にプロバイダに接続してWebページが表示できれば常時接続環境が実現できたことになる。

この状態で外部からIPアドレスによる接続は可能になっているはずだ。たとえば、Webサーバが起動していれば、Webブラウザのアドレスに「111.122.133.130」などと指定すればホームページにアクセスすることができる。

次は独自ドメインを取得しよう。

画面9　「Routing and gateways」での設定




```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:40:95:30:15:7F
          inet addr:111.122.133.130  Bcast:111.122.133.255 Mask:255.255.255.248
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:931 errors:0 dropped:0 overruns:0 frame:0
          TX packets:412 errors:0 dropped:0 overruns:0 carrier:0
          collisions:1 txqueuelen:100
          Interrupt:11 Base address:0xc000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:48 errors:0 dropped:0 overruns:0 frame:0
          TX packets:48 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

IPアドレス　ネットマスク

画面10　ifconfigコマンドによるIPアドレスの確認

ISDN フレッツ 常時接続
ドメインを取る！

常時接続の環境が整ったら、いよいよドメインを取得しよう。

ドメインについてはいまさら説明するまでもないと思うが、日本国内向けのJPドメインや、米国で使用されているCOM、NETドメインなどがある。

JPドメインは、ccTLD（Country Code Top Level Domain：国別トップレベルドメイン）と呼ばれ、JPNIC（日本ネットワークインフォメーションセンター）が管理している。

COM、NET、ORGドメインは、gTLD（Generic Top Level Domain）と呼ばれ、NSIが管理している。そのほかにも、各国のccTLDもあり、UK（イギリス）や、最近流行の「TO（トンガ）」、「CX（クリスマス諸島）」などがある。

本来、ccTLDはその国向けのドメインなのだが、現在では各国で自由に登録できるようにしているドメイン管理組織が多い。COMドメインなどのgTLDも、米国内向けのドメインであったのだが、各国で自由に登録することができる。



画面11　WORLDNIC-JPのホームページ（gTLDレジストラ）

一方、日本のJPドメインは、世界で最も厳しい条件が規定されていて、簡単に取得することはできない（**コラム**参照）。ドメインの取得条件が厳しいうえ、ドメインの申し込みに稼働中のDNSが必要となるなど、手続きが面倒だ。そこで、編集部では、取得が容易なgTLDを取得することにした。

なお、ドメインは一般的に先着順に登録される。つまり、先に申請したほうが使用する権利がある。当然ではあるが、先に取られたドメインは使用することができない（ただし、所有権や商標権などで、ドメイン使用が差し止められることはある）。

ドメインを取得するには、各国のドメイン管理組織か、レジストラと呼ばれるドメイン取得代行を行っている会社に依頼することになる。

今回は、日本国内向けgTLDドメインのレジストラである,WORLDNIC-JPで取得することにした。WORLDNIC-JPでは、gTLDである、COM、NET、ORGドメインの申請を日本語で行うことができる。また、国内にサポート体制があるなど安心できる点も嬉しい。ただし、料金の支払いにはクレジットカードが必要となる。もっとも、レジストラのほとんどは、料金の支払いにクレジットカードが必要になるので、WORLDNIC-JPが特別だというわけではない。そういわけで、クレジットカードを用意しておこう。なお、ドメイン名登録料は1万2800円（2年間）だ。

WORLDNIC-JPによるドメイン申請手順

WORLDNIC-JPでのドメインの取得は簡単だ。ホームページ（http://www.worldnic-jp.com/）にアクセスして、「ドメイン名の登録サービス」を選択する。次に、「ドメイン名検索」を選択しよう。

使用できるサブドメイン名は、英数字（小文字）とハイフンの組み合せで、3～22文字までだ。ただし、サブドメイン名の先頭と最後にハイフンを含めることはできない。以上の組み合せでドメインを考えてみよう。

編集部としては、「linuxmagazine.com」というドメインを取得したいところだ。早速、「linuxmagazine」というサブドメインを入力して検索してみる。が、COM、NET、ORGともに取得されていた（残念）。そこで、気を取り直して「ascii-linux」を検索してみることにした。なんと、COM、NET、ORGすべてのドメインが取得可能ではないか。

ならば、やっぱりCOMがいいということで、「ascii-linux.com」で取得することにする。

あとは、画面のメッセージに従って、必要事項を入力していくだけだ。

登録にあたって、いくつか注意点をあげておく。

①個人／法人による登録

法人と個人の場合で登録画面が異なるが、法人の場合は法人名と連絡担当

者、支払い担当者を記入する必要があるだけで、ほかは変わらない。

②パスワード

パスワードは2種類必要になる。1つは、会員管理情報の「アカウントパスワード」で、これは登録情報の変更など、WORLDNIC-JPでのサービスを受けるためのものだ。もう1つは、「WorldNICパスワード」で、DNSなどの設定を変更する場合に必要となるものだ。

③DNS

DNS（Domain Name System）に関しての情報も入力しなければならない。のちほど設定する予定のネームサーバ情報を記入しておく。DNSのIPアドレスはサーバのIPアドレスとなるので、こちらは問題ないとしても、ホスト名を決定しておかなければならない。もちろん、自由に設定して構わないのだが、DNSとして相応しい名前を付けたほうがいいだろう。たとえば、ns1などがいいだろう。また、DNSにはバックアップ用のセカンダリDNSも必要だ。セカンダリはプライマリDNSと自動的に同期して、バックアップサーバとして稼働する。セカンダリに関しては、別途サーバを用意してもいいが、セカンダリの本来の目的であるバックアップという点から考えると、別のネットワークに設置するべきだろう。編集部では、プロバイダにセカンダリを依頼することにした。

サンプルのネットワーク計画では、**表2**のような情報となる。

ドメインの取得までには数日かかるので、ドメインが登録されるまでにDNSを設置しておこう。

ネームサーバの設置

独自ドメインによる運営を行うには、必ずネームサーバ（DNS）を設置しなければならない。それも、ネットワーク内（LAN）向けのネームサーバではなく、インターネット側へ公開するネームサーバだ。

このネームサーバは重要で、これによってインターネット上のマシンから、特定のWebサーバなどの名前の解決ができるようになる。

ネームサーバの設定は非常に複雑なので、ここではサーバ公開に必要な、

プライマリホストアドレス	111.122.133.130
プライマリホスト名	ns1.ascii-linux.com
セカンダリホストアドレス	プロバイダのネームサーバのアドレス
セカンダリホスト名	プロバイダのネームサーバのホスト名

表2　ドメイン申請に必要なネットワークの情報

Column

日本語ドメイン

最近のニュースでも話題になったが、ドメイン名に日本語が登録できるようになった。いままでのドメインの、英数字と「－（ハイフン）」の組み合せだけでなく、たとえば「リナックスマガジン.com」といった表記が利用できるようになる。

COMドメインなどのgTLDを管理しているNSIなどでは、すでにマルチバイトのドメイン登録を開始している。

ただし、JPドメインでの登録はまだ開始されていない。

日本のJPドメインを管理しているJPNICは、非常に厳格なポリシーにのっとり、1組織1ドメインの原則を貫いてきた。しかし、これでは企業の商品名といったドメインは取得しづらいし、個人で取得するのは困難だ（実際には、セカンドレベルに「gr」を割り当てたグループドメインや、地域型ドメインを利用するしかない）。そこで、日本語によるドメインを運用するにあたり、今までのセカンドレベルを付加する形式とは別の、汎用ドメインが導入されることになった。汎用ドメインでは、「jp」の前にいきなり好きな名前を付けることができ、誰でもいくつでも登録することができる。これは、現状の「co」などのセカンドレベルとは別のものとして割り当てられ、従来のセカンドレベルは厳格なルールのまま運用されることになる。

もともと漢字を母国語とする国では、マルチバイトによるドメインの表記を希望する声はあったが、インターネットという世界標準の上においては、マルチバイトを使用することに否定的だった。しかし、米国のベンチャー企業であるi-DNSが、いきなり台湾でマルチバイトによるドメインサービスを開始してしまったのだ。このi-DNSのサービスは「インターネットの世界が崩壊する」と大騒ぎとなってしまった。幸いにも、この件によって、マルチバイトによるドメインの実現方法が正式に検討され、マルチバイトドメインが使用可能になったのだ。

JPNICでは、今年1月22日から「優先登録申請期間」として日本語汎用ドメインの登録を開始する。この期間は、ドメインの所有者を対象としたそのドメインの日本語ドメインの登録と、商標権などを持つ保有者のみが登録できる。それ以外の人が登録できるのは3月1日からとなる。

しかし、実際に日本語ドメインが現状で使えるのかというと、実はそういうことではない。現状では、Webブラウザや、メーラの対応が必要になる可能性が高い。また、現在はテスト期間であり、マルチバイトドメインが問題ないかどうかが検証されている最中だ。マルチバイトドメインを実現する仕組みも、ドメインを管理する団体によって異なっており、取得したドメインが将来にわたって使えるかどうか保証はない。ということで、現状で実際に運用するのであれば、マルチバイトドメインは避けたほうがいいだろう。

最低限の設定だけを解説する。ネームサーバの詳しい解説は、次回以降にすることにしよう。

まず、DNSの簡単な仕組みをおさらいしておこう。

DNSは、コンピュータの名前を解決するための仕組みだ。インターネット上のコンピュータは、IPアドレスによって管理されている。このアドレスは、111.122.133.144といった数字のアドレスで表現されるもので、人間にとっては覚えにくい。そこで、このアドレスに、www.ascii-linux.comといった人間が覚えやすいアドレスを割り当てて、DNSによってこれら2つのアドレスの変換を行うようにしているのだ。

このドメイン名から数字のIPアドレスへの変換を正引き（順引き）、数字のIPアドレスからドメイン名への変換を逆引きという。通常、人間が使うのは正引きだが、コンピュータにとっては逆引きも重要である。

Linuxや多くのシステムでは、BIND（Berkeley Internet Name Domain）というプログラムを使ってDNSを構築することができる。ここでは、現在多く使われているBIND 8を例に設定方法を紹介することにしよう。

BIND 8では、BINDの設定ファイルと、マシンのアドレス情報を記載したいくつかのゾーンファイル（DNSデータベースファイル）を作成する必要がある。設定ファイルがいくつかに分かれているのも、BINDを複雑にしている理由のひとつだ。

BIND 8の設定ファイルは次のようなものだ。

```
/etc/named.conf
    （BIND設定ファイル）
/var/named/local.ca
    （ルートキャッシュファイル）
/var/named/domain
    （ゾーンファイル（正引き））
/var/named/domain.rev
    （ゾーンファイル（逆引き））
/var/named/domain.local
    （ローカルゾーンファイル）
```

このうち、named.conf以外は、named.conf内で定義するので、実はどんなファイル名でも構わない。今回は上記のようなファイル名にしたが、慣例的に正引きにはドメイン名を、逆引きにはIPアドレス（ネットワーク部）を付けることが多い（ascii-linux.comと111.122.133といったもの）。ネットワークが複雑になった場合、このような名前を付けておけば、混乱が少ないからだ。

それでは順に設定をしていこう。ただし、ここでの設定はインターネットに公開される重要なものなので、十分注意してすること。設定を間違えると、ほかのネットワークに迷惑をかけてしまう可能性があるからだ。

■named.conf

まずは、BIND設定ファイルであるnamed.confからだ（**リスト1**）。

ここでの設定は、ゾーン（サブネットと考えてよい）の設定と、ゾーンファイルの指定だ。

最初にある「options」セクションでは、ゾーンファイルが置かれているディレクトリを指定している。

4つある「zone」では、各ゾーンについて設定している。

最初のゾーン「.」は、ルートDNS

▼リスト1　BIND設定ファイル（/etc/named.conf）

```
options {
        directory "/var/named";
};
```
ゾーンファイルのディレクトリの指定

```
//ルートキャッシュファイル
zone "." {
        type hint;
        file "named.ca";
};
```
ルートDNSの設定

```
// 正引きゾーン
zone "ascii-linux.com" {
        type master;
        file "domain";
        notify yes;
};
```
正引きゾーンの設定

```
// 逆引きゾーン
zone "SUB128.133.122.111.in-addr.arpa" {
        type master;
        file "domain.rev";
        notify yes;
};
```
逆引きゾーンの設定

```
// ローカル
zone "0.0.127.in-addr.arpa"{
        type master;
        file "domain.local";
};
```
ローカルループバックの設定

だ。これは最も上位のDNSを指定しているルートキャッシュファイルだ。

次のゾーンである、「ascii-linux.com」は、自分自身のドメインのゾーンを設定している。

その次の「SUB128.133.122.111」は、自分自身のIPアドレスのゾーンを設定している。すぐに気付くと思うが、IPアドレスを逆から記述した形式になっている。「SUB128」に関しては、ちょっと特殊な記述で、DNSの解説書などには載っていない。

実は、BINDではクラスC未満の小さなネットワークのゾーンを指定することはできない。クラスCとは、最後の8ビットがホスト部になるネットワークのことである（ネットマスク255.255.255.0、111.122.133.000～111.122.133.255など）。クラスによるネットワークの分割は、すでに過去のもので、サンプルのネットワーク計画のように、現在ではCIDR（Classless Inter-Domain Routing）という、ビット単位での分割が一般的になっている。BINDでは、編集部で申し込んだBiz IP8のような3ビットのサブネットなどは設定することができないのだ。これを解決するために、プロバイダではクラスCに別名を付けて、サブネット化してユーザーに割り当てている。このサブネット名がSUBxxxとなっているのだ。このサブネットの名前の付け方は、プロバイダによって異なる。また、ユーザーによっても異なるので、プロバイダから提供されたサブネット情報に合わせて設定する必要がある。Biz IP8では、このサブネットがSUB128（128は割り当てられた最初のIPアドレスの最後の8ビット）といった名前になっている。

named.confの最後のゾーンは、ローカルループバックの設定だ。

各行のセミコロンは必要なので忘れないように注意すること（本当によく忘れる）。

■local.ca

このファイルはルートドメインのネームサーバ情報が保存されているルートキャッシュデータだ。といっても、実際はキャッシュファイルではなく、ルートドメインのネームサーバのアドレスが記述されているだけだ。このファイルは自分で作成するのではなく、最新のファイルをインターネット上から入手する。また、不定期に更新されるので、常に最新のものをチェックしておく必要がある。実際にはファイルをftp://rs.internic.net/domain/named.rootなどで入手してリネームする。

■domain

このファイルは正引き用のゾーンファイルだ（**リスト2**）。ドメイン名からIPアドレスへの変換をするための情報を書き込むためのものだ。

最初の行のは、SOA（Start Of Authority）レコードと呼ばれるもので、ゾーンを定義している。「@」はそのゾーン自体を示す置換記号だ。「SOA」のうしろの最初のフィールド「ns1.ascii-linux.com.」は、プライマリネームサーバの名前、その次のフィールド「rina.hogehoge.co.jp.」は管理者のメールアドレスである。いずれの場合も最後に「.」が付いていることに注意しよう。これは、アドレスをFQDN（Fully Qualified Domain Name）で記述する必要があるからだ。また、メールアドレスは「@」部分を「.」として記述する。このメールアドレスは、DNSに問題があった場合などにそのネームサーバの管理者と連絡をとるための重要なものだ。

次の行はシリアル番号と呼ばれる、ゾーンファイルの版番号だ。このシリアル番号はどんなものでも構わない。しかし、上位のネームサーバは、このシリアル番号をチェックしてゾーンファイルが更新されたかどうかを確認する。このため、ゾーンファイルに変更を加えた場合は、このシリアル番号を必ず変更しなければならない。この番号は、重複しないように日付＋版番号などとしておくのが一般的だ。

その次の行からの4つの数値、「Refresh」「Retry」「Expire」「Minimum」は、上位のDNSや、ほかのDNSがゾーン情報をどのように扱うかの指定だ（**表3**）。

とりあえずは、**リスト2**の数値のま

▼リスト2　正引きゾーンファイル（/var/named/domain）

```
@       IN      SOA     ns1.ascii-linux.com. rina.hogehoge.co.jp.  (
                                2000122501 ; Serial      ← シリアル番号
                                28800      ; Refresh
                                14400      ; Retry
                                3600000    ; Expire
                                86400 )    ; Minimum

        IN NS   ns1.ascii-linux.com.
        IN NS   ns2.sphere.ad.jp.          ← ネームサーバの指定

ns1     IN A    111.122.133.130            ← nsi.ascii-linux.com=111.122.133.130

www             IN CNAME        ns1.ascii-linux.com.   ← wwwでも参照

localhost       IN A 127.0.0.1  ;ループバックアドレスの定義
```

まで構わないだろう。

その次の「IN NS」は、ネームサーバの指定である。ここにプライマリネームサーバとセカンダリのネームサーバの名前を記述する。FQDNで記述されていることに注意すること。この2つの行の先頭が空白になっているのは、省略した記述方法で、実際にはゾーン自体が暗黙で指定される。簡単にいえば、省略した場合は、SOAレコードで指定した「@」行が引用される。

サンプルのネットワーク計画では、セカンダリをプロバイダに依頼することにしているので、セカンダリにはプロバイダのネームサーバを指定している。

次の「ns1」で始まる行は、プライマリネームサーバのIPアドレスを指定している。ns1というマシンが、111.122.133.130であると指定している。

「www」で始まる行は、別名の定義である。ns1というマシンはwwwという別名を持つという指定だ。つまり、ns1でも、wwwでも接続できるということだ。

最後の「localhost.」は、ループバックアドレスの定義だ。ここにだけ「.」が付いていることに注意しよう。「.」を付けないと「@」行の引用が使われてしまうからだ。

■domain.rev

このファイルは逆引き用のゾーンファイルだ（**リスト3**）。前述した正引き用のゾーンファイルとほとんど同じ構造だということがわかるだろう。最後の「130」から始まる行が、IPアドレスを指定している部分だ。ここでは、130（111.122.133.130）というIPアドレスが、ns1.ascii-linux.comというマシンだということを指定している。

■domain.local

このファイルはローカルループバックの定義をしているファイルだ。記述方法は正引きゾーンファイルとほぼ同じなのでわかるだろう。ここでは、127.0.0.1がlocalhostだと定義をしている。

リゾルバの設定

ネームサーバのゾーンファイルの記述がすんだら、ネームサーバの指定を行う。これをリゾルバ設定という。DNSのゾーンファイルの設定に比べれば、とても簡単なので名前で惑わされる必要はない。ようは、DNSサーバの名前を指定するだけだ。

リゾルバの設定は、/etc/resolv.confファイルを設定することで行う（**リスト5**）。

最初の行はドメイン名を指定する。ここではドメイン名をFQDNで指定していない（「.」が必要ない）ことに注意しよう。残りの2行はネームサーバのIPアドレスだ。最初がプライマリネームサーバ、2つめがセカンダリネームサーバとなる。これだけだ。

ネームサーバのテスト

ネームサーバの設定ができたら、DNSを起動する前に、もう一度設定ファイルやゾーンファイルの内容を確認してみよう。セミコロンを忘れていたり、ピリオドを忘れていることはよくあるし、前述したように設定値が間違

数値	意味
Refresh	セカンダリネームサーバがプライマリネームサーバを参照する間隔
Retry	プライマリネームサーバの参照に失敗した場合のリトライ間隔
Expire	プライマリネームサーバが死んでいると判断するまでの間隔
Minimum	ほかのネームサーバがゾーン情報を保持する間隔

表3　ゾーン情報関連の間隔（単位は秒）

▼リスト3　逆引きゾーンファイル（/var/named/domain.rev）

```
@       IN SOA ns1.ascii-linux.com. rina.hogehoge.co.jp. (
                                        2000122501 ; Serial
                                        28800      ; Refresh
                                        14400      ; Retry
                                        3600000    ; Expire
                                        86400 )    ; Minimum

        IN NS   ns1.ascii-linux.com.
        IN NS   ns2.sphere.ad.jp.

130     IN PTR  ns1.ascii-linux.com.
```

ネームサーバの指定

111.122.133.130=nsi.ascii-linux.com

リスト4　ローカルゾーンファイル（/var/named/domain.local）

```
@       IN      SOA     localhost. root.localhost.  (
                                        1997022700 ; Serial
                                        28800      ; Refresh
                                        14400      ; Retry
                                        3600000    ; Expire
                                        86400 )    ; Minimum
        IN      NS      localhost.

1       IN      PTR     localhost.
```

リスト5　リゾルバの設定（/etc/resolv.conf）

```
domain ascii-linux.com
nameserver 111.122.133.130
nameserver 123.123.123.123
```

っていた場合、ほかのネットワークに迷惑をかける可能性があるからだ。十分に注意して内容を確認してからDNSを起動してほしい。

問題がなければ、次のようにしてDNSを起動する。

```
# /etc/rc.d/init.d/named start
```

または、

```
# /etc/init.d/named start
```

設定ファイルやゾーンファイルを変更した場合は、「start」の代わりに「restart」とすることで設定値を有効にすることができる。ゾーンファイルを変更した場合は、シリアル値を変更することを忘れないようにしよう。

DNSが起動したら、名前の解決ができるようになったはずだ。名前の解決が正常に行われているかどうかは、nslookupコマンドで確認することができる。

```
# nslookup ns1.ascii-linux.com
```



画面12　ntsysvコマンドによる、namedデーモンを自動的に起動させるための設定

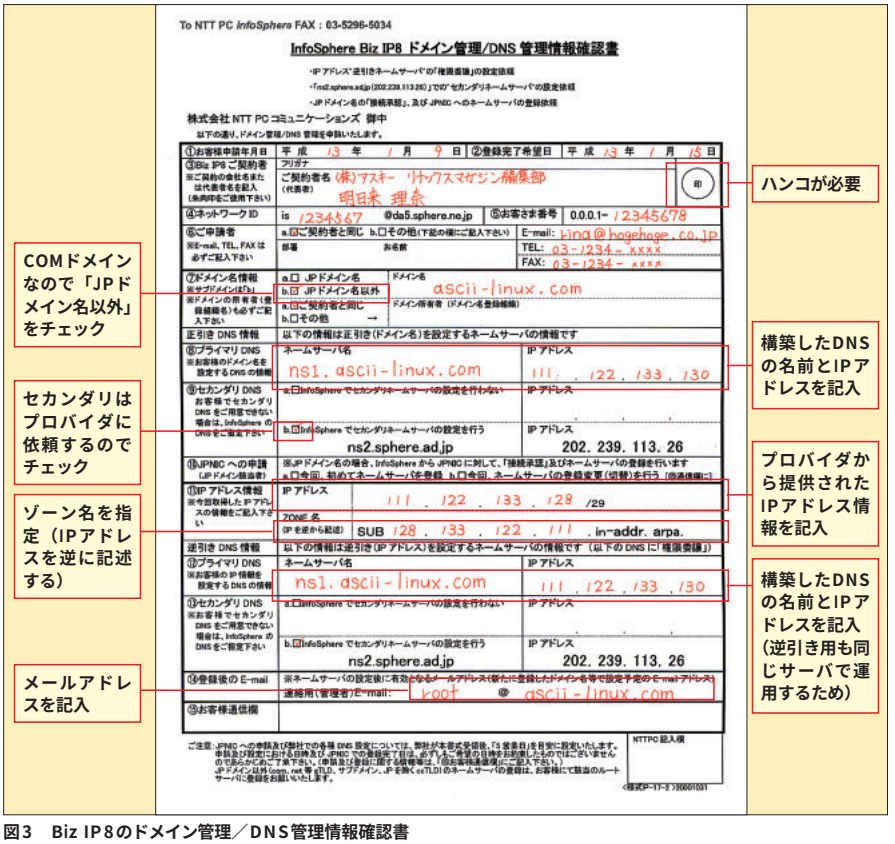


図3　Biz IP8のドメイン管理／DNS管理情報確認書

正常に名前の解決ができていれば、正しいIPアドレスが表示される。これは正引きの例だったが、IPアドレスを指定することで逆引きをテストすることもできる。

```
# nslookup 111.122.133.130
```

しかし、たぶん、逆引きでは名前の解決は正常に行われないだろう。これは、前述したCIDRによるサブネットの影響だ。通常であれば、自分自身のIPアドレスなので解決できると思えるが、このIPアドレスは実は自分自身のゾーン（SUB128.133.122.111）に属しているとは見なされず、プロバイダのDNSまで問い合わせが行われてしまう。しかし、プロバイダのDNSにはまだ、このサブネットの情報が登録されていないために、名前の解決ができないのだ。

これを解決するためには、プロバイダにゾーンの設定を依頼する必要がある。Biz IP8では、「ドメイン管理／DNS管理情報確認書」を提出することになる（**図3**）。

プロバイダでサブネット情報の登録が完了すれば、逆引きも正常に行われるようになる。

なお、正常に名前の解決ができない場合は、設定ファイルやゾーンファイルをもう一度見直してほしい。/var/log/messagesにDNS（named）を起動した際のメッセージログが記録されているので、これを参照してみるといいだろう。

DNSが正常に設定できたら、ntsysvコマンド（TurboLinuxではturboservice）コマンドで、namedがシステム起動時に有効になるように設定しておこう（**画面12**）。これで、次回起動時には自動的にDNSが起動する。

セキュリティ

ここまであえてセキュリティについては触れてこなかったが、フレッツ・ISDNによる接続だとはいえ、常時接続であればセキュリティについて十分注意する必要がある。固定グローバルIPアドレスはいつでもサーバを特定でき、24時間アタック可能だとなればなおさらだ。

どうせ大したデータもないし、アタックされても構わないなどと思うかもしれないが、これは大きな間違いだ。確かに、仮にサーバが乗っ取られたとしても実害は少ないかもしれないが、このサーバを踏み台として、ほかのマシンへのアタックに利用されるかもしれないのだ。あるいは、あなたのサーバを不正中継して、スパムメールを大量に送られたとしたらどうだろうか。あなたは世界中から非難を受けることになるだろう。このようなことが起きないためにも、セキュリティを十分考慮する必要がある。

さて、ここでサーバのセキュリティをどのように構築するかを解説したいところだが、セキュリティについてはとても今回だけで解説することはできない。そこで、詳しいセキュリティに関しては次回以降にするとして、今回はとりあえずダイヤルアップルータを使ったファイアウォールを構築しておこう。これだけでも効果はある。

ダイヤルアップルータのほとんどには、IPパケットフィルタリングと呼ばれるセキュリティ機能が用意されている。これは、いわゆるフィルタ機能で、IPパケットやポートを制限することでLANのファイアウォールを実現するものだ。

IPパケットフィルタリングもかなり複雑で、解説するとかなりの分量になる。そこで、今回はDNSとHTTPだけを通すようなフィルタリングの設定方法を紹介しておく。

今回編集部で利用した、RTA50iは、WebブラウザでIPパケットフィルタリングを設定できる。ダイヤルアップルータに接続して、「IPフィルタの設定」ページで設定する。

ここで、ポート番号53番と80番（DNSとHTTP）だけを外部に公開し、それ以外のポートは通信を拒否するように設定しておこう。IPパケットフィルタは、リスト番号の若い順に評価され、条件にマッチするとそのフィルタが実行されるようになっている。つまり、最初にポート53番か80番をまずチェックし、それ以外のポートへの接続は拒否するように設定すればいい。これらの番号は別名が割り当てられており、domain、wwwといった名前で設定することもできる。

設定したIPパケットフィルタは画面13のようになっている。

RTA50i以外のダイヤルアップルータでも、基本的には同じなのでマニュアルをよく参照して設定してほしい。多くのダイヤルアップルータでは、

```
ip filter 80 pass * * * 53,80 *
ip filter 81 pass * * * * 53,80
ip filter 82 reject * * * * *
```

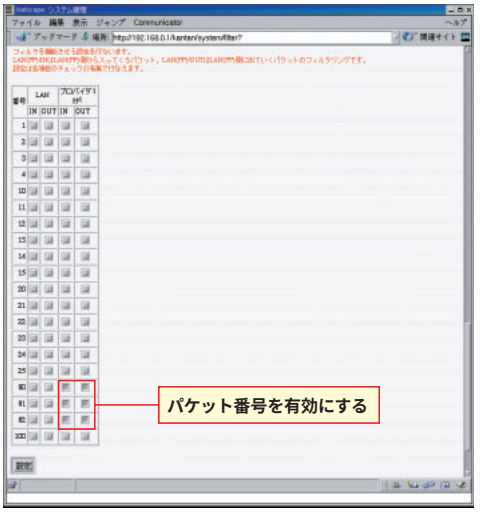
といったIPパケットフィルタを設定すればよい。この例では、フィルタ番号53番と80番にマッチしないパケットはすべて接続を拒否することを意味している。

IPパケットフィルタを設定したら、インターネット側への入出力時に有効になるようにフィルタ番号を有効にする（**画面14**）。

これでDNSとHTTP以外のポートへの接続が拒否されるので、とりあえずのセキュリティとして機能させることができる。



画面13　RTA50iでのパケットフィルタリング（53番と80番）



画面14　パケットフィルタの適用





Crusoe
で夢見るモバイルな日々

文：Linux magazineラボ
Text : Linux magazine Lab.



「いつかはスーパーモバイラーに」

いつの頃からか、そんな思いを抱えている。常にネットにアクセスし、絶えず情報を発信しつづけるサイバーエリートに対する憧れがあるからだ。しかし現実は厳しくて……。
そんなある日のこと、わが編集部にSONYのC1VJがやってきた。噂のCrusoeマシンである。その魅力にすっかりまいってしまった私に、「Crusoeを調査せよ」との指令が下された。再び頭をもたげるスーパーモバイラーへの野望を胸に秘め、さっそく調査にとりかかった。



Crusoeとは何者なのか？

2000年のコンピュータ業界を振り返ってみるとき、Crusoeの登場は最大の話題のひとつであったといえると思う。クロック数によって象徴的に表わされるCPUの処理性能ではなく、省電力性能で勝負するというTransmetaの戦略は新鮮な驚きをもって迎えられた。

省電力を実現するしくみ

では、Crusoeはどのようなしくみで電源消費を低減しているのだろうか？3つのキーワードがある。「VLIW」、「CMS」、そして「LongRun」である。

●VLIW

Very Long Instruction Wordの略で、CPUアーキテクチャのひとつ。CrusoeのCPUコアで採用されている。「極長命令セット」と訳されるように、1つの長い命令セットの中に複数の命令をまとめて、同時に（1クロックサイクルで）多くの命令を処理する。複数の命令を同時に処理することで、処理効率を高めようという考え方だ。

並列処理によって効率を高めるという考え方自体はPentiumやAthlonでも採用されている。ただしこれらのCPUでは、受け取った命令を解析し、同時に処理可能な命令を判断して命令セットとしてまとめ、その順序を決定しながら動作するという流れになる。これに対して、VLIWアーキテクチャのCPUは、あらかじめ決められた一定数の命令を受け取った順に処理していく。そのため比較的、内部構造が単純で、トランジスタ数も少なくて済むことから、消費電力を抑えることが可能となる。

●CMS

受け取った命令セットを単純に同時実行するというVLIWアーキテクチャの基本特性は、受け取った命令が同時に実行されても大丈夫だという前提がないと成立しない。「誰か」が整合性のとれた命令セットを作成してCPUに渡す必要があるのだ。PentiumやAthlonではCPU内部で（実装された回路で）これらの判断を下している。Crusoeでこの役割を果たしているのがCMS（Code Morphing Software）だ。

CMSには、もうひとつの大きな役目がある。内部アーキテクチャの異なるCrusoeのCPUコアは、x86命令を解析できない。そのままではx86互換のソフトウェアは動作しないのである。CMSはCrusoeマシンが起動しているあいだバックグラウンドで常に動作し、OSから渡されるx86命令をCrusoeネイティブの命令セットに変換してCPUコアに受け渡す。CMSはCrusoeがx86互換となるためのコアコンポーネントなのだ。

●LongRun

CPUのクロック周波数と内部電圧を多段階に動的に変更することで、消費電力を低減するテクノロジー。単純に言うと、システムで動作しているソフトウェアの実行状況を監視して、CPUパワーがそれほど必要ないときには、クロックと内部電圧を下げて電力消費を抑えるしくみだ。最適な動作クロックと電圧はCMSが判断している。

700MHz動作のTM5400では、200～700MHzの範囲で16段階の切り替えが行われるとされている。200MHz以下ではCPUの動作が保証できないため、従来のCPUと同じくスリープモードに移行することで電力消費を抑える。

これまでCPUの内部にハードウェアとして実装されていた機能を、CPUの外部で、しかもソフトウェアで実現することによって、CPUコアの構造を単純化している点がCrusoeの最大の特徴だ。これはベースにある「低消費電力のCPUを作る」という設計思想から生まれたアイデアなのだろう。

Column

LinuxとCrusoe

TransmetaにLinus Torvalds氏が所属しているためかどうかわからないが、Linuxカーネルの次期安定バージョンである2.4.0のテストソースには早い時期からCrusoeの最適化コードが含まれていた。CPUタイプとして「CONFIG_MCRUSOE」という項目があるのがその選択オプションだ。

次ページ以降で紹介する今回のテストでも、CPUタイプに「Crusoe」を選択したカーネルでCrusoeマシンを動作させたが、処理スピードなどの面で、それ以外のオプションを選択したカーネルとの有意な差は感じられなかった。

また、現在のところCrusoeの持つLongRun機能を有効に活用するためのしくみはLinuxには用意されていない。このため、ほかのCPUを搭載したマシンとのバッテリ駆動時間の大きな違いはないと思われる。カーネルの最適化、ロングラン性能の向上とも、今後のコミュニティの成果を期待して待つことにしよう。

Linuxをインストールしてみたぞ！

Crusoeマシンといっても結局はx86互換なのだから、Linuxが動かないわけがない。とはいえ、Linusさんも開発に携わったCPUでもあるし、Linux magazine的にはテストしないわけにいかないでしょう。ということで、Linuxの動作テストを敢行することにあいなったのである。

今回テストしたのは、SONYのVAIO C1VJ/BP、富士通のFMV-BIBLO LOOX S5/53、そしてNECのLaVie MXの3機種。まずは、動作保証外であるLinuxのテストに機材を貸与してくださったNEC様へ「どうもありがとうございました」。ちなみにVAIOは本誌編集部・デスクUの私物、LOOXは編集部で購入したものだ。最後になりましたが、日立様および日立ファンの皆様へ「Priusなくて申し訳ないです」。

テストマシンのスペックについてざっと触れておこう。VAIOとLaVieはCrusoe TM5600-600MHz、LOOXがTM5400-533MHzを搭載。標準のメモリ容量は3機種とも128Mバイトだが、テストに使用したマシンでは、VAIOのみメモリを追加しトータル192Mバイトとなっている。各マシンのスペックに関しては、解説のなかでも適宜紹介する。また103ページ以降に、各マシンの詳しいスペック表を掲載してあるので参考にしてほしい。



画面1　Windows Meのデバイスマネージャ

ノートPCにインストールするコツ

さっきも言ったように、Crusoeはx86互換で動作するので、特別なインストールの注意事項があるわけではない。一般的なノートPCでのインストール方法を踏襲すればいいはずだ。

まずはマシンのハードウェア構成を把握しておくことが重要。インストールする前に、I/Oコントローラ、ビデオ、サウンドなどのチップの種類を確認しておこう。これらの情報はマニュアルやメーカーのWebサイトに掲載されているが、最も確実で簡単なのがWindowsのデバイスマネージャ（**画面1**）を使う方法。チップ名はもちろん、リソースの使用状況なども確認できる。

このためにも、プリインストールされているWindowsは、きちんとLinuxでの動作が確認できるまで消さずに残しておくほうが無難である。

ハードウェア情報の収集が終わったら、インストール作業へと進むわけだが、ここで問題となるのがブートメディア。インストールCD-ROMからブートできて、そのCD-ROMがインストールドライブとして認識されれば一番いい。そのままデスクトップマシンと同様にインストールできる。

CD-ROMドライブからブートできない場合には、フロッピードライブからブートしてインストーラが立ち上がるか確認してみよう（ブートフロッピーについては**コラム**を参照）。フロッピーブートできれば、ハードディスクまたはネットワーク上のファイルを利用してインストールを行うことが可能だ。

今回のテストでは3台のマシンで、

Column

ブートフロッピーの作り方

通常、ディストリビューションの商用パッケージにはインストール用のブートフロッピーが付属しているが、ノートPCの場合、PCMCIAサポートが含まれたカーネルイメージが必要となることが多い。また、ネットワークインストール用のブートイメージが必要なこともある。

このようなケースでは、ブートフロッピーを別途作成しなければならない。商用パッケージ、FTP版にかかわらず、インストール時のブート用にいくつかのカーネルイメージが用意されているので、そのイメージを使ってブートフロッピーを作成しよう。Linuxマシンで作成するには、インストールCD-ROMをマウントして次のコマンドを実行すればいい。

```
# cd /mnt/cdrom/images
# dd if=boot.img of=/dev/fd0
```

カーネルイメージが収録されているディレクトリとイメージファイルの名称（上記の例では/images/boot.img）については、ディストリビューション付属のドキュメントに記載されているはずだ。

Windowsマシンで作成するにはrawriteコマンドを使う。このコマンドもインストールCD-ROMに収録されている。DOSプロンプトで実行し、メッセージにしたがってイメージファイル名とカーネルイメージを書き出すドライブを指定すればいい。



別々の方法を試みた。それぞれについて簡単に説明しておこう。テストは、すべてKondara MNU/Linux 2000を使用して行っている。

■CD-ROMドライブからのインストール

B5サイズ以下のミニノートPCの場合、内蔵CD-ROMドライブを持たないことが多いが、外付けのCD-ROMドライブでもPCカード経由の接続であれば、インストールドライブとして使用できる可能性がある。

VAIO C1VJの場合、純正のCD-ROMドライブPGCA-CD51はPCカード接続だ（**写真1**）。この純正ドライブはインストールに使用できる。ただし、起動時にカーネルオプションを指定する必要がある。boot:プロンプトで以下のように指定すればOKだ。

```
boot: text ide2=0x180,0x386
```

「ide2」は2つめのIDEコントローラとして接続されることを示し、それに続く16進数の指定はドライブが使用するI/Oポートアドレスを示している。ブートラベルに「text」と指定しているのは、Anacondaが正常に起動しないためだ（テキストベースのインストーラを使うことになる）。あとは通常の手順でインストールできる。

■ハードディスクからのインストール

ディストリビューションによっては、ハードディスクにコピーされたファイルからのインストールが可能だ。Kondara 2000は、このインストール方法をサポートしている。LaVie MXとKondaraの組み合わせで実際に行ったインストールを例に説明しよう。

まず、Linuxをインストールするパーティション以外のハードディスク領域にインストールCD-ROMの/RedHat/baseと/RedHat/RPMS以下のファイルをすべてディレクトリごとコピーする。Linuxはまだインストールされていないので、当然コピー操作はWindowsで行う。特別に領域を確保する必要はなく、WindowsのFATパーティションでかまわない。

ブートにはフロッピードライブを使った。LaVie MXではUSB接続のサードパーティ製ドライブ（矢野電器のUFD-04）にブートフロッピーをセットして起動できた。ブートデバイスについては、マシンによってBIOSで設定が必要な場合があるかもしれない。このあたりはマニュアルで確認してほしい。

LaVie MXの場合も、テキストインストーラを使用した。インストールメディアの指定で、「ハードディスク」を選択する以外は通常のインストール手順と同様だ。

■ネットワークインストール

パーティションに十分な空きがない場合やLinuxパーティションのみでマシンを使用したい場合には、ネットワーク経由でのインストールが有効だ。特にNFSを利用したインストールは、設定も簡単、かつネットワーク上のCD-ROMドライブを直接マウントできるのでお勧めである。LOOXとKondaraで実行した手順を紹介しよう。

前提としてネットワーク環境が必要（あたりまえ）だ。インストール側のネットワークカードには、プラネックス・コミュニケーションズのENW-3503-Tを使用した。もう1台NFSサーバとなるLinuxマシンが必要となる。基本的なLANの設定は適切に行われていることにして話を進めよう。まず、NFSサーバとなるマシンの設定から。インストールCDをサーバにマウントし



写真1　SONY PCGA-CD51
VAIOノートの純正CD-ROMドライブ。接続用のPCカードは「NinjaATA」と認識され、LinuxのPCMCIAサービスでサポートされている。

ておく。次に/etc/exportsファイルに次の1行を追加する。

```
/mnt/cdrom (ro,no_root_squash)
```

そして、NFSの起動スクリプト/etc/rc.d/init.d/nfsにrestartオプションを付けて実行する。これでネットワーク上のすべてのマシンに対してNFSサーバの/mnt/cdromディレクトリが公開される。

インストール側の設定として、まずはブートフロッピーを作成する。これはネットワークカードの認識にPCMCIAのサポートを含んだカーネルイメージが必要なためだ。Kondaraの場合は、インストールCD-ROMの/images/pcmcia.imgがそれにあたる。

次に作成したフロッピーを使ってインストーラを起動するわけだが、ここでもLaVie MXのときと同じUSBフロッピードライブを使用した。なおLOOXの場合、起動時に「Fujitsu」のロゴ画面でF12キーを押すとブートドライブを選択できる。

インストーラはやはりテキストベースのほうを選択する（boot:プロンプトで「text」を指定）。インストールドライブの指定画面で「NFS image」を選択し、次に表示される画面でNFSサーバのIPアドレス（またはホスト名）

Crusoe
で夢見るモバイルな日々

とインストール元となるディレクトリ「/mnt/cdrom」を指定する。正しくNFSが設定されていて、ネットワークカードの認識がうまくいっていれば次の画面に進み、以降通常のインストール手順となる。

●

インストールに成功したら、次はやはりXやサウンドを設定して、使用環境を充実させたい。設定に関しては機種ごとに説明することにしよう。

大苦戦だよVAIO C1VJ

純正のCD-ROMドライブを使って無事インストールが完了した優秀なVAIO君だったが、設定に関してはキカン坊であった。XFree86、サウンドともにかなりの設定手順を踏まなければ動いてくれないのだ。以下、そのあたりを紹介していく。

■Xの設定

VAIO C1VJに搭載されているビデオチップはATIのRage Mobility M1だ。XFree86 3.3.6では、Mobility MおよびPはSVGAサーバでサポートされているが、M1は未サポートであるようだ。「1」が付いただけなのに残念なことである。フレームバッファを有効にすれば、標準のSVGAサーバで動作するが、Xの終了後にコンソールに復帰できないなどの問題点があるようだ。

実はこのあたりの情報は、オーナーであるデスクUからもたらされたものである。そして、Xの設定に関する耳寄りな情報が新たにデスクUから寄せられた。金城さんという方が自身のWebサイトでMach64サーバに対するパッチを公開しているというのだ（http://www05.u-page.so-net.ne.jp/tc4/t-kinjo/vaio/）。さっそく、このパッチをゲット＆適用して、Xサーバをリコンパイルしてみたところ、バッチグーであった。金城さんありがとう！

手軽なバイナリバージョンも用意されているので、そちらの導入手順を紹介しておこう。Xサーバがインストールされているディレクトリにコピーして、起動用のシンボリックリンクを作成するだけでいい。たとえばKondaraであれば、以下の手順になる。

```
# cd <ダウンロード先>
# gzip -d XF86_Mach64_C1VJ.gz
# mv ./XF86_Mach64_C1VJ /usr/X11R6/bin/
# ln -sf /usr/X11R6/bin/XF86_Mach64_C1VJ /etc/X11/X
```

XF86ConfigのMonitorとScreenセクションを**リスト1**に示すので参考にしてほしい（金城さんのWebページにもサンプルがあります）。色深度に関しては、Displayサブセクションの「Depth 16」の部分を変更すれば、24bpp、32bppでの表示も可能だ。

■サウンドの設定

サウンドに関してもVAIO君はキカン坊だった。Xの設定に成功して、気分良くalsaconfで設定を行ったところ、「ビャゴーー！ガガッ！ビィーーーー！」としか表記しようのない音がとてつもない大音量で鳴り響き始めたのだ。よその編集部からは失笑を買うは、となりに座ってるデスクU@VAIOオーナー様は「こわさないでー」と真っ青になるはで大変だったんである。

事前に行ったWindowsでの調査では、C1VJのサウンドコントローラは「YAMAHA DS-XG PCI Audio」と表示されていた。ALSAのドライバはymfpciを使用、サウンドチップはYAMAHAのYMF754と認識されていた。設定は間違ってないと思いつつALSAのWebサイト（http://alsa.linux.or.jp/）を確認したところ、最新の0.5.9dでymfpciドライバに修正が加えられているようなので、とりあえずバージョンアップすることにした。

新しいドライバでは、異音は発生しなかったが、今度はウンともスンとも言わなくなった（mpg123とXMMSを使ってテスト）。conf.modulesの設定は**リスト2**のとおり。ALSAのインストールドキュメントを見る限り、設定に間違いはない。XMMSの出力プラグインを変更してALSAにダイレクトに出力したり、Esound経由で試したりしてみたが、やはりだめ。

デバッグモードでALSAのドライバ

▼リスト1

```
Section "Monitor"
    Identifier   "LCD panel Ultra-wide 1024x480"
    HorizSync    31.5 - 95.0
    VertRefresh  40-150
    Modeline "1024x480"  60 1024 1024 1053 1072  480  485  492  525
EndSection

Section "Screen"
    Driver      "accel"
    Device      "ATI Rage Mobility M1"
    Monitor     "LCD panel Ultra-wide 1024x480"
    Subsection "Display"
        Depth       16
        Modes       "1024x480"
    EndSubsection
EndSection
```



をコンパイルし直してログメッセージを確認してみると、何やらIRQかDMAの問題であるらしい。PCIコントローラがうまく競合を解決できていないのかもしれない。

そこで、一縷の望みをかけてカーネルのバージョンを上げてみることにした。カーネル2.4ではPCIまわりが改良されたという記述をどこかで見たような記憶があったからだ。ALSAは2.4のテストカーネル（2.4.0-testxx）にも対応しているので、ALSAの側にカーネルのバージョンに関する問題はないはずである。

カーネルをバージョンアップして、ALSAをリコンパイル、モジュールをロードする。alsamixerでボリュームを調整して、mpg123でテスト。若干雑音が混じるものの鳴っている。続いてxmmsでもテスト、こちらはOSSドライバ、Esound経由ともに、雑音なくきれいに再生できた。やれやれである。原因として考えられるのは、やはりPCIまわりなのか？　よーわからんわたしではあった。

▼リスト2

```
alias char-major-116 snd
alias snd-card-0 snd-card-ymfpci
alias char-major-14 soundcore
alias sound-slot-0 snd-card-0
alias sound-service-0-0 snd-mixer-oss
alias sound-service-0-1 snd-seq-oss
alias sound-service-0-3 snd-pcm-oss
alias sound-service-0-8 snd-seq-oss
alias sound-service-0-12 snd-pcm-oss
options snd snd_major=116 snd_cards_limit=1 snd_device_mode=0666
snd_device_gid=0 snd_device_uid=0
options snd-card-ymfpci snd_index=0 snd_id=CARD_0
snd_dac_frame_size=128 snd_adc_frame_size=128 snd_ac97_frame_size=128
```

LOOXも結構……

さて、DVD-ROMドライブもH"INモジュールもないモデルをしたLOOXはどうであったか？　結論からお伝えしておくと、C1VJよりは設定はスムーズにいくが、それなりに手間は必要といったところである。順を追って説明しよう。

■Xの設定

LOOXでは、今回テストしたSシリーズ、上位モデルのTシリーズとも、ビデオチップにATI Rage Mobility Mを採用している。VAIOのところで少し触れたように、このチップはXFree86 3.3.6のSVGAサーバでサポートされている。1024×512という特殊な解像度を持つディスプレイを正しく設定できれば、Xが動作するはずである。

実際に、Xconfiguratorで適当な解像度を設定し、以下のコマンドで動作状況を確認したところ、Modelineの解決以外はどうやら動いているようだ。

```
# startx -- -probeonly &> x.log
# less x.log
```

問題はModelineの設定のみと考え、解決策を探っていると、スタッフより

Column

MORTION EYEも使える！

VAIO C1VJのウリのひとつがディスプレイの上部に内蔵されたビデオカメラ「MOTION EYE」だ。当初はLinuxでは使えないものとあきらめていたのだが、検索サイトであれこれ調べているうちにVAIOの内蔵カメラをLinuxで動作させる専用ツールがあるという情報が得られた。

調査してみると、もともとはC1XS用に作成されたツールであることがわかった。ダメもとで、ダウンロード（配布元のURLはhttp://samba.org/picturebook/）して試したところ、C1VJでもちゃんと動作した。JPEGフォーマットの静止画とMJPEGによるAVIフォーマットの動画がキャプチャできる。操作もWindowsと同様に「CAPTURE」ボタンを使って行えるようになっている。

Webページに掲載されている作者のコメントによると、まだ試作段階とのことだが、VAIOの機能をフルに活用したい方は試してみてはいかがだろうか。



実行中のMOTION EYEツール





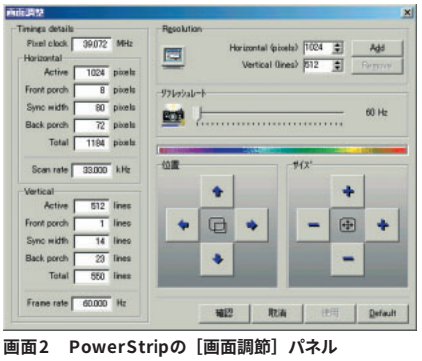
これがMOTION EYEだ！（写真上）CAPUTUREボタンとジョグダイヤルで操作するのだ（写真下）

有力な情報を得ることができた。「PowerStrip」というWindows用ツールを使えば、Modelineの元になるディスプレイの情報をゲットできるというのだ。すばらしい！

PowerStripはhttp://www.entechtaiwan.com/ps.htmから入手できる。シェアウェアなので、ライセンス条件に関しては付属のドキュメントをよく読んでほしい。

Modelineの指定に必要な情報は、PowerStripメニューの［アドバンスオプション］－［画面調整］で表示される（**画面2**）。左側に並んだ数値に注目してほしい。［Pixel clock］がModelineに設定する最初の数値にあたる。水平周波数は［Horizontal］に表示されている数値を元に求められる。アクティブなピクセル数が1024、これはModelineの2番めに指定する数値。［Front porch］が「8」ピクセルとなっているので、1024＋8で1032が3番め。さらに［Sync width］の「80」を足して4番めには1131を指定する。［Total］はそのまま5番めの数値だ。水平解像度も同様に［Vertical］の値を元に求められる（**リスト3**を参照）。

ホントにこれでいいの？　と突っ込まれると、はっきりいって「動物的な勘」的なところもあるのだが、でも動いたから、たぶん合っているのだろう。わたしの勘もバカにできないのだ。



画面2　PowerStripの［画面調節］パネル
Linuxのxvidtuneにあたる機能だ。検出される数値もほぼ同じもの。

■サウンドの設定

サウンドに関しては、富士通のWebページに載っていたALiのドライバをまず試してみた。ダウンロードサイトのURLはhttp://www.ali.com.tw/eng/support/drvlnx.htm、ファイル名はaudio-auto2.2.x_2.3.35_2.tar.gzだ。これは名前からもわかるように2.2.xから2.3.35までのカーネルに適用するパッチになっている。それ以降のバージョンのカーネルソースには、このコードが組み込まれているようだ（2.4.0-testxxも含む）。

パッチの適用とカーネルモジュールのコンパイル手順は付属のREADMEに詳しく記載されているので、そちらを参考にしてほしい。コンパイルとインストールが完了したら、モジュールをロードしよう。モジュールファイルはali_m1553.o、ロードするコマンドは「modprobe ali_m1553」だ。今回のテストでは一発でOKだった。

カーネル2.4.0の場合、カーネルコンフィギュレーションのSoundオプションで「Trident 4DWave DX/NX, SiS 7018 or ALi 5451 PCI Audio Core」を選択する（詳しくはターボリナックスジャパンのWebページが参考になる。URLはhttp://www.turbolinux.co.jp/knowledge/public/340.html）。モジュールファイル名はtrident.oになる。こちらも、今回テストした範囲では結果は良好であった。以上でLOOXの設定も完了。結構疲れた〜。

LaVie MXは優等生？

今回の3機種の中では唯一ジャストB5サイズのノートPCである。ディスプレイの縦横比も通常の4:3で、最大解像度は800×600。ということで、Xの設定は特に問題なし。Xconfiguratorで「Extended Super VGA 800×600」を選択し、一発で起動できた（グラフィックチップはLOOXと同じRage Mobility M）。

サウンドも、搭載されているチップESS 1946S（solo-1E）がALSA 0.5.8でサポートされているので、Kondaraの場合はalsaconfコマンドを使って設定できた。ただし、設定直後はマイクのインプットレベルが最大になるため、ハウリングが起こる。もちろんこれはalsamixerコマンドで音量を調整すればおさまるのだが、結構大きな音がするのでビックリするかもしれない。

内蔵のLANインターフェイスもLinuxで利用可能だ。コントロールチップはInetl i82559。インストーラで認識され、conf.modulesに自動的に設定が追加された（EtherExpress Pro100のドライバで動作）。接続もOK。

と、ここまでは非常に優秀なLaVie MXだったのだが、実はここまで伏せていた重大な問題があったのだ。それもまあ納得できなくもない理由によるものではあるのだが、詳しくは製品紹介のコーナーで……。

▼リスト3

```
Section "Monitor"
    Identifier  "My Monitor"
    VendorName  "Fujitsu"
    ModelName   "LOOX"
    HorizSync   30-60
    VertRefresh 50-100
    Modeline "1024x512" 39.026 1024 1024 1111 1184  512 513 526 550 -
HSync -VSync
EndSection
```



SONY

VAIO C1VJ

オープンプライス（実売価格20万円前後）
http://www.vaio.sony.co.jp/



VAIOシリーズに共通のスタイリッシュなデザインが素敵だ。コンパクトな筐体にMOTION EYE、ジョグダイヤル、IEEE 1394、USB、メモリスティックにオーディオビジュアル出力といった多くの機能が詰め込まれている魅力的なマシンだ。うわさではLinusさんも、C1VJの北米仕様であるC1VNを購入したらしい。

テストで使用したのはC1VJ/BPというモデルだったが、これは外付けCD-ROMドライブとMicrosoft Office 2000 Personalをバンドルしたもの。本体の仕様はまったく共通である。

もともと世界的に人気の高いシリーズだが、Linuxerにも熱狂的なファンが多いようで、本文やコラムで紹介したようにXサーバのパッチやMOTION EYE用のツールの開発といったユーザーによる積極的な取り組みが進められている。今回は試せなかったが、XFree86 3.3.6のSVGAサーバ用のパッチ、XFree86 4.0.1のドライバモジュール用のパッチなども、すでに有志の手によって開発されている模様だ。

また、Webサイトで情報を公開しているオーナーやVAIOユーザーを対象としたメーリングリストなど、ネット上での情報交換もさかんである。これらは、LinuxでVAIOを使いたい方には、うれしいポイントだろう。

Linuxで使用する際には問題点もある。ディスプレイが最高輝度に設定されて変更できないようなのだ。Windowsでの動作時にはジョグダイヤルを使って設定を変更できるが、ジョグダイヤルが使えないLinuxからは変更不可なのである。

最後に、1024×480ピクセルという変則なスクリーンサイズについて、実際に使ってみた感想を。GNOMEコントロールセンターやNetscapeの設定画面で下段に表示されるボタンが隠れてしまうといった細かな突っ込みどころはあるものの、慣れてしまえば特に問題はないように感じられた。先ほども指摘したように最大輝度なので異常に明るいのだが、これとて暗いよりはまし。コントラストの具合がいい感じで、文字がクッキリ、ハッキリ見える点は文句なくgood。ちなみにディスプレイの輝度と同じくジョグダイヤルによる音量の変更はできない。

VAIO C1VJの主なスペック	
CPU	Transmeta Crusoe TM5600 600MHz
サウスブリッジ	Intel 82371AB
システムバスクロック	100MHz
メモリ	128Mバイト（最大192Mバイト）
ビデオチップ	ATI RAGE Mobility M1（VRAM容量8Mバイト）
サウンドチップ	YAMAHA DX-XG（YMF754）
LCD	1024×480ピクセルTFTカラー液晶
ハードディスク	約12Gバイト（約8Gバイト/約4Gバイト）
サイズ（W×D×H）／重量	248mm×152mm×27mm／980g
その他の装備	USBポート×1、内蔵ビデオカメラ（6インチ35万画素CCD）、TypeIIPCカードスロット（CardBus対応）×1、メモリスティックスロット（マジックゲート対応）×1、内蔵モデム（V.90/K56Flex自動対応ソフトウェアモデム）など



写真2　C1VJの左側面
IEEE1394ポートとPCカードスロット、メモリスロット（PCカードスロットの下）が配置されている。写真右端にあるのはエアダクト（？）だ。



写真3　C1VJの右側面
写真左側から、モデム、外部モニタ、USB、マイク、ヘッドフォン、オーディオビデオと外部インターフェイスがズラリと並ぶ。

Crusoe
で夢見るモバイルな日々

富士通
FMV-BIBLO LOOX S5/53

オープンプライス（実売価格16万円前後）
http://www.fmvworld.net/



今回テストしたLOOX S5/53は、DVD-ROMドライブもH"（エッジ）INモジュールも搭載していない最もシンプルなモデルだ。外部インターフェイスは、PCカードスロット（TypeII）、USBポート、内臓モデムのモジュラジャック、サウンドのヘッドフォンアウトとマイクインといたってシンプルである。

LOOX Sシリーズには、もうひとつH"INモジュールを標準搭載したS5/53Wがある。また、DVD-ROMドライブ内蔵のTシリーズがあり、やはりH"INモジュール搭載のT5/53WとH"INなしのT5/53の2機種がラインナップされている。Tシリーズは、Crusoe TM5600 533MHzを搭載し、ディスプレイがひとまわり大きく（Sシリーズは8.8型、Tシリーズは10型）、解像度も1280×600と高くなっている。

実際に使ってみた感想を少し。ディスプレイの視認性とキーボードのタッチは、どちらも許容範囲。ふだんノートPCを使っていないので、最初は少しとまどったが、しばらくいじっているうちに気にならなくなった。細かい点だが、ヘッドフォン出力のミニピンジャックは側面にあったほうが使いやすいのではと感じた。外部インターフェイスが少ないのだから、その取りまわしには気を使ってほしいところだ（と思います＞富士通さん）。

使っていて便利だったのが、ディスプレイの下に取り付けられた液晶インジゲータだ（**写真4**）。バッテリの残量、ACアダプタやPCカードの着脱が表示され、OSに依存しないでハードウェアの状態を知ることができるので、特にLinuxで動作しているときに重宝した。VAIO C1VJではできなかったLinux動作中のファンクションキーによるディスプレイの輝度とサウンドボリュームの調整についても、LOOXはどちらも可能だ。ディスプレイの表示範囲も32ライン縦に広いぶん使いやすい。ただしコントラストの具合はVAIOのほうが良好なように感じた。



写真4　液晶インジケータパネル
バッテリの残量がひと目でわかるのがうれしい。

LOOX S5/53の主なスペック	
CPU	Transmeta Crusoe TM5400 533MHz
サウスブリッジ	ALi M1535
システムバスクロック	133MHz
メモリ	128Mバイト
ビデオチップ	ATI RAGE Mobility （VRAM容量4Mバイト）
サウンドチップ	ALi 5451 PCI Audio Core（M1535ビルトイン）
LCD	1152×512ピクセルTFTカラー液晶
ハードディスク	約10Gバイト（約4.8Gバイト/約5.2Gバイト）
サイズ（W×D×H）／重量	243mm×151mm×30mm／980g
その他の装備	USBポート×1、TypeIIPCカードスロット（CardBus対応）×1、内蔵モデム（V.90/K56Flex自動対応ソフトウェアモデム）、携帯電話接続USBケーブルなど

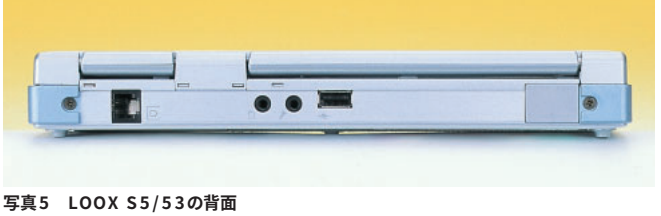


写真5　LOOX S5/53の背面
バッテリはキーボードの手前に装着されているので、背面に外部インターフェイスを配置している。液晶パネルを開いた状態では、やや使いづらい。



写真6　LOOX S5/53の右側面
PCカードスロット、ACアダプタのコネクタはこちら。写真左端のボタンは「メール」ボタン。もちろんLinuxでは使えない。



NEC

LaVie MX LX60T/51EC

オープンプライス（実売価格24万円前後）
http://121ware.com/



LaVie MXの最大の特徴は、そのバッテリ容量にある。液晶パネルと本体の蝶番部分にある通常のバッテリ以外に液晶パネルの内部にリチウムポリマーバッテリが組み込まれているのだ。カタログ上の最大バッテリ駆動時間はなんと11時間！　もちろん使用条件によって変わってくるのだろうが、それにしても驚異的である。

バッテリ駆動を延ばすための工夫としては、反射型液晶パネルの採用もあげられる。反射型液晶は、ほかの多くのノートPCで採用されているバックライト付きの液晶よりも、バックライトがないぶん電力消費が少なくて済む。そして、もちろんCrusoeも省電力設計に貢献している。

これらの仕様からわかるように、Lavie MXはバッテリによるロングラン性能を基本コンセプトとしている。つまり、Crusoeマシンだからロングランなのではなく、「ロングランを実現するためのCrusoeの採用」とういことだ。

さて本文中で予告した問題点について。省電力のために採用された反射型液晶パネルが暗いのである。通常のオフィスのように室内で照明のみの明かりしかない環境では、かなりきついレベルである。デスクライトなど補助的な照明を必要かもしれない。ディスプレイの輝度がどの程度必要なのか、用途によっては購入の際の重要な判断ポイントになるだろう。個人的には、モバイル用途でも明るい日中の屋外で使用することはあまりないような気がするのだが……。なお、これはLinux、Windowsに共通の問題である。

Linux使用時に気になったのが、バッテリ残量が確認できない点だ。VAIOとLOOXは、電源管理機能としてACPI（Advanced Configuration and Power Inteface）とあわせてAPM（Advanced Power Management）が実装されており、apmコマンドやGNOMEのバッテリアプレットで残量を確認できた。LOOXでは液晶インジゲータでも確認可能だ。LaVie MXはACPIのみのサポートである。LinuxではACPIがサポートされていない（カーネル2.4でサポートされるが、管理ツールなどは現状では未整備）ため、ツールやコマンドを使って確認することはできない。Linuxで使用する際にはバッテリ残量に注意するようにしたい。

LaVie MXの主なスペック	
CPU	Transmeta Crusoe TM5600 600MHz
サウスブリッジ	ALi M1535
システムバスクロック	120MHz
メモリ	128Mバイト（最大192Mバイト）
ビデオチップ	ATI RAGE Mobility M（VRAM容量4Mバイト）
サウンドチップ	ESS Technology Solo-1E（ESS1946S）
LCD	800×600ピクセル反射型TFTカラー液晶
ハードディスク	約20Gバイト
サイズ（W×D×H）／重量	264mm×211mm×21～27mm／約1.37kg
その他の装備	USBポート×2、TypeIIPCカードスロット（CardBus対応）×1、内蔵モデム（V.90/K56Flex自動対応ソフトウェアモデム）、内蔵LANインターフェイス（100BASE-TX/10BASE-T）など



写真2　C1VJの左側面
IEEE1394ポートとPCカードスロット、メモリスロット（PCカードスロットの下）が配置されている。写真右端にあるのはエアダクト（？）だ。



写真3　C1VJの右側面
写真左側から、モデム、外部モニタ、USB、マイク、ヘッドフォン、オーディオビデオと外部インターフェイスがズラリと並ぶ。

挑戦! OpenBlocksサーバ化計画

文：渡邉利和
Text : Toshikazu Watanabe
(toshi-w@tt.rim.or.jp)



ぷらっとホームのOpenBlockSは、体積にしてタバコの箱4つ分くらいの小型Linuxサーバだ。標準ではNATとパケットフィルタリングが使える専用アプライアンス機として利用するのが普通だ。だが、せっかくのLinuxマシン、もっといろいろな利用法があるだろう。そこで、ストレージを追加してグレードアップし、静粛、コンパクトという利点を活かした使い道を考えてみた。

OpenBlockSは、イーサネットポートを2つ備えた超小型Linuxサーバである。といっても、標準構成ではストレージとして本体内蔵の4Mバイトの Flash ROMがあるだけなので、実体としては「シェルが使える専用ルータ」といった感じのハードウェアである。このため、基本的には組み込みOSとしてLinuxが使われている、と見るのが正しく、PCでLinuxを運用するのと同様の環境が簡単に手に入るということではない。

とはいえ、超小型の筐体と究極の静音設計は魅力的であり、その特徴に合った使い道を考えるのが正しい姿勢といえるだろう。そこで今回は、Compact Flashやハードディスクを増設して自由度を増して、どんな使い道が開けるか挑戦してみた。

ファームウェアのアップデート

さて、作業に取りかかる前にまずファームウェアが最新版かどうか確認しておこう。本稿執筆時点での最新ファームウェアはVer.1.09iというバージョンだ。

OpenBlockSの場合、ファームウェアというのはPCでいうところのBIOS、と思ってしまってはいけない。このファームウェアにはOSとして利用するLinuxそのものが含まれているのだ。そのため、ファームウェアのアップデートは実際にはディストリビューションのバージョンアップくらいに相当すると考えたほうがよい。特に、最新版であるVer.1.09iではコマンドラインで利用できるスクリーンエディタaeeが含まれているので、OpenBlockS上で設定ファイルの書き換えなどが簡単にできるようになり、使い勝手がかなり良くなっているのでお勧めだ。

では、アップデート作業の概要を紹介しておこう。まずは、ぷらっとホームのWebページから新バージョンのファームウェアのファイルを入手する。ファイルサイズは約3.7Mバイトだ。OS全体なのでそれなりのサイズになっているが、PCに普通にディストリビューションをインストールするのに比べれば驚くほどのコンパクトさだといえる。

さて、このファイルをOpenBlockSに転送するには、WindowsまたはLinuxが動作しているPCを転送元として利用し、製品に添付されている専用コマンドを使用して行う。Windows用はバイナリでそのまま実行できる形式だが、Linux用の専用コマンドはソースで提供され、ユーザーがそれぞれコンパイルして利用することになる。Windows用のコマンドはWebサイトに用意してあるので、ダウンロードすることも可能だ。

なお、OpenBlockSのIPアドレスはデフォルトで192.168.253.254/24（eth0側）に設定されており、ファームウェアのアップデートの際にはこのアドレスでアクセスすることになるので、PC側でこのIPアドレスに合わせてやる必

要がある。また、ファームウェアのアップデートの際には、OpenBlockSの筐体を開けて内部のディップスイッチを変更する必要がある。ごく小さなスイッチで操作しにくいので、慎重に行おう。

必要な準備が完了したら、転送元のPCでコマンドを起動し、入手したファームウェアファイルを指定して転送する。今回はWindowsマシンから転送した（**画面1**）。転送は数分で終了するので、済んだらOpenBlockSの電源を切り、ディップスイッチの設定を元に戻せば完了だ。

ファームウェアのアップデート作業の詳細に関しては、http://openlab.plathome.co.jp/OpenBlockS/に詳しく解説したページが用意されているので、そちらも参照していただきたい。ファームウェアの入手もここからできる。

Compact Flashの追加

OpenBlockSには、追加ストレージとしてCompact Flashと2.5インチハードディスクを内蔵できるようになっている。これら追加ストレージを利用すれば、標準では動作していなかったDNSやsendmail、wu-ftpdが使えるようになるため、サーバとしての活用の幅が広がる。これらの追加機能は、標準ではメモリが足りないので動作していないのだ。追加ストレージを用意すれば、あらかじめ用意してあるこれらのサーバ機能が利用できるので、活用しない手はない。

Compact Flashは、32Mバイト以上の容量のものがサポートされている。もちろん、大きければ大きいほどストレージの容量が拡大できることになるが、あまり欲張っても大変だろう。また、ハードディスクを増設すればGバイト単位でストレージが増え、本格的なサーバらしくなってくるが、ハードディスクには可動部分があるため、どうしても動作音は無音とはいかないし、動作中に衝撃を与えたりすればクラッシュの危険もあるので、多少使い勝手が変わってくる面もある。

では、設定だ。実のところ、物理的な設定はごく単純で、筐体を開けると目の前に用意されているスロットにCompact Flashを差し込むだけである。この状態で電源を入れると新たにLEDが点灯し、正しく認識されていることが確認できる。

ソフトウェア的な設定については、設定スクリプトが用意されているので、ログインしてこれを実行するだけで済む。ただし、このスクリプトでは16Mバイト＋16Mバイトの計32Mバイトしか利用しない。16Mバイトはスワップに使われ、残る16Mバイトは/ext1にマウントされる。ここに追加コマンドが収録されているわけだ。

今回使用したCompact Flashは64Mバイトなので、あと32Mバイトが丸々手つかずで残っている。もったいないので、これは手作業でパーティションを設定し、適当にマウントして利用することにしよう。手順に関しては、標準で用意されている設定スクリプト（/etc/mkext1など）が参考になる。



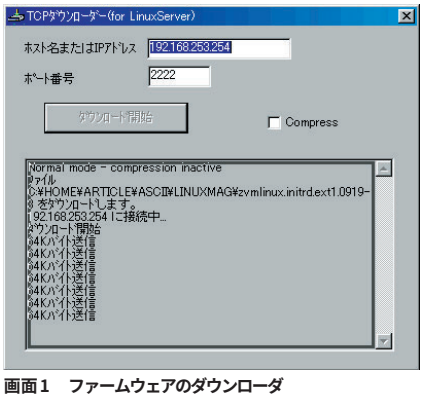
写真1
手のひらサイズのLinuxサーバだ。

作業としてはごく単純で、fdiskコマンドを使って追加のパーティションを作成する。次いで、/sbin/mke2fsコマンドを利用してパーティションをフォーマットし、mountコマンドで任意のマウントポイントにマウントしてやればよい。設定ユーティリティなどが充実している最近のディストリビューションとは異なり、このレベルの作業はすべてコマンドラインから行う必要があるが、基本的な理屈がわかっていればそう難しいものではない。

追加ストレージからの起動

さて、実は追加ストレージとして導入したCompact FlashやハードディスクからOpenBlockSを起動するように設定することも可能だ。

OpenBlockSはシステムを本体内蔵のフラッシュメモリに格納しており、起動時にはRAMディスクを作成してルートファイルシステムのイメージをコピーし、そこから起動する、という方式を採っている。フラッシュメモリのイメージは、ユーザーが簡単に変更できるようにはなっていないため、ミスオペレーションに関しても安全性が高い。たとえ設定を滅茶苦茶にしてしまったとしても、リブートすれば元通りに回復されるので安心だ。



画面1　ファームウェアのダウンローダ
ファームウェア（Linux）はWindows／Linuxマシンから送り込むことができる。

反面、この方式にはユーザーが意図的に行った変更であっても、リブートすると失われてしまう、という欠点もある。これを防ぐために/usr/sbin/flashcfgコマンドが用意されている。このコマンドは、引数で指定されたリストファイルに従って、内蔵フラッシュメモリにファイルを保存する。

```
# flashcfg s /etc/flashcfg
```

などのように利用する。“s”はSaveの意味でファイルの保存動作、/etc/flashcfgは標準で使われているリストファイルである。このファイルの中を見ると、ファイルのパス名が1行1ファイルの形で列挙されている。このリストに挙げられたファイルの変更であれば、上記のコマンドラインを単に実行するだけで、その変更はリブート後にも保存されるようになる。なお、リストにないファイルの変更を保存したいのであれば、単にリストファイルのほうを書き換えてファイル名を追加すればよい。

標準の環境で、設定ファイルに多少手を加える程度で運用するのであればこれでよいが、新しいコマンドをインストールするなどの作業を行い、あちこちのディレクトリに複数のファイルを新しく置いた、などという場合にはリストファイルに必要な変更を正しく反映させるのは面倒になってくる。この場合、いっそ追加ストレージからシステムをブートするようにしてしまえば作業はぐっと簡単になるはずだ。この場合は内蔵Compact Flashからのファイルの復元は行われず、ルートファイルシステムは追加ストレージ上のファイルシステムがそのまま使われることになるためだ。

Compact Flashもハードディスクもどちらも起動デバイスとして利用できるほか、マニュアルを見る限りでは両者を併用して好きなデバイスからシステムを起動することも可能なようだ。ただし、今回の試用では成功しなかった。デバイスの相性があるのか、設定が悪いのか不明だが、Compact Flashとハードディスクの両方を接続するとシステムが起動しなくなってしまうのだ。ただ、現実の運用を考えればこの両者を併用する必要性は薄いだろう。今どきハードディスクはGバイト単位であり、数十Mバイト程度のCompact Flashを併用しなくてもストレージとしての容量に不足はないだろう。

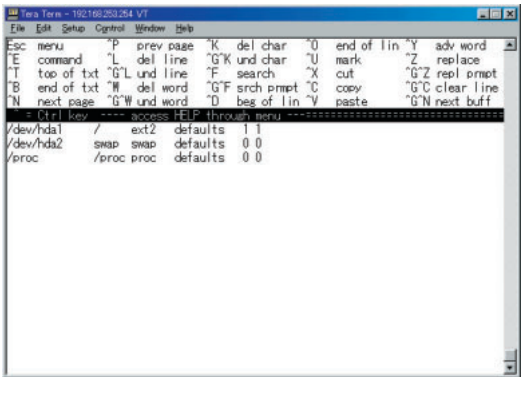
さて、追加ストレージを起動デバイスとするためには、追加ストレージ上にルートファイルシステムを構築する必要がある。標準では/etcディレクトリにmkcfrt、mkhdrtの2つのスクリプトが用意されており、これを利用するだけでよい。ただし、パーティションをデフォルトで2つ作成し、1つをファイルシステム、1つをスワップにするのはmkext1と同様で、パーティションサイズもmkcfrtではmkext1と同様に16Mバイトずつとなっていた。今回利用したCompact Flashは前述のとおり64Mバイトのものだが、あまり細かくパーティションを分割しても使いにくいので、実際には48Mバイト＋16Mバイトに分割し、48Mバイトのパーティションをルートファイルシステムにしている。mkcfrtを使うとこうした任意のサイズのパーティション設定はできないので、mkcfrtスクリプトの中身を書き換え、パーティションの作成処理を省略した別コマンドを作成し、あらかじめ手作業で作成しておいたパーティションにルートファイルシステムのコピーだけ実行するようにしてみた。

作業自体はごく簡単なことだが、実際に行ううえではOpenBlockS上でテキストエディタが使えないとなかなか面倒なことになる。ここで効いてくるのがVer.1.09iで提供されたaeeである（**画面2**）。このエディタが標準で利用できるようになり、OpenBlockSを運用するのは格段に楽になった。

さて、新しいルートファイルシステムが用意できたら、本体のディップスイッチを切り替えて再起動すればCompact Flash（またはハードディスク）からシステムが起動するようにな



写真2
Compact Flashは拡張インターフェイス基板の上にセットされる。



画面2　スクリーンエディタaee
新しいファームウェアではスクリーンエディタが追加され、設定が容易になった。

る。ホームディレクトリにファイルを保存しておく、といった使い方をするのであれば、追加ストレージを用意してホームディレクトリを追加ストレージ上の別パーティションに移動しておくか、あるいは追加ストレージから起動するようにして運用しないとリブートの際にすべて失われるのでくれぐれも注意していただきたい。

基本的な設定

さて、実は少々回り道をしたが、OpenBlockSの基本的な設定はWebブラウザ経由でネットワーク上の任意のシステムから簡単に実行できる（**画面3**）。しかも、Web経由で設定変更した結果はシステムが自動的に保存し、Compact Flashへの待避という面倒な作業については一切考える必要もないので、通常はこちらを利用するのが賢明だろう。なお、Web設定画面を開くためにはユーザーIDとパスワードによる認証が行われるほか、2つあるイーサネットポートのうち、LAN側で利用する10/100BASE-TXポート側からしかアクセスできないようになっているので、セキュリティ面でもあまり心配はない。では、基本的な設定を行っていこう。

まずはネットワーク設定である。画面左側のメニュー一覧から「ネットワーク設定」をクリックすると、さらに3つのサブメニューが開く。「イーサネット1の設定」「イーサネット2の設定」「ネットワークの設定」である。ここで、イーサネット1（eth0）は10/100BASE-TXインターフェイスでLAN側、イーサネット2（eth1）は10BASE-TでWAN側である。

なお、このときファームウェアをアップデートしたことが活きてくる。新しいファームウェアではeth1をDHCPクライアントとして設定できるため、プロバイダなどから割り当てられるネットワーク設定をそのまま反映させることができるようになるのだ。OpenBlockSをDHCPクライアントとするためには、/etcディレクトリにあるdhclient-script1を起動すればよい。

さて、インターフェイスごとに適切なTCP/IP設定を行い、「ネットワーク設定」でホスト名やドメイン名、デフォルトゲートウェイの設定を行えば、基本的なIPルータとして動作するようになる。ルーティングテーブルを手動で設定するためには[ルーティング]－[コマンドの入力]、NAT（IPマスカレード）を利用するためには、[アドレス変換]－[コマンドの入力]メニューで必要なコマンドラインを入力してやればよい。

IPルータ／簡易ファイアウォールとしてOpenBlockS

OpenBlockSは、家庭内で個人がIPルータとして利用するには理想的なデバイスである。簡単なファイアウォール機能を設定するなど、機能の追加が容易に行えることに加え、小さいので設置場所に困ることもない。さらに、ファンがないため動作音がまったくしないのも家庭向きである。

筆者宅もそうなのだが、PCがたくさんあってもそのすべてを常時運転しているわけではない。というのも、音と消費電力が気になるためだ。そうはいってもインターネットに常時接続されている環境であれば、少なくともルータくらいは常時運転していないと都合が悪いので、現在筆者宅ではルータ役を担っているPCは24時間連続稼働中である。しかし、夜などは周辺が静かなせいもあり、PCの電源ファンの音や、ときどき聞こえてくるハードディスクのシーク音などが妙に気になるものである。本人はまあ許容範囲だと思っていても、家族からは大変評判が悪い。このルータをOpenBlockSに置き換えれば、夜の静寂を妨げることもなくなり、たいそう具合が良いはずだ。

また、標準状態ならもちろん、Compact Flashを追加してストレージを増量しても、機械的な可動部分はゼロのままである。このため、稼働中でも設置場所に気を遣う心配はあまりなく、稼働中にちょっと移動させるくらいのことも問題なく行える。外形もコンパクトで軽量なので、持ち歩いて利用するデバイスとしても便利だろう。

常時接続の普及に伴い、インターネットに接続されたPCのセキュリティが問題になっているが、Windows PCを直接接続するのに比べ、LinuxベースのOpenBlockSは基本的なセキュリティレベルは高い。そこで、持ち歩きが簡単という特徴を活かすと、ノートPC用の外付けセキュリティBOXという使い方もあり得るだろう。自宅でもオフィスでも、ノートPCをインターネットに接続する際には常にOpenBlockSを経由するようにするわけだ。

OpenBlockSはDHCPサーバ／クライアントになれるため、こうした接続



画面3　Webインターフェイスによる設定
本来はWeb画面による簡単設定が身上だ。

を行う場合はノートPCの側の設定は一切変更する必要がないこともメリットになるだろう。自宅とオフィスのネットワーク環境の違いなども、OpenBlockSに吸収させることが可能になる。かつ、IPマスカレード機能を活用すればインターネット側から直接ノートPCにアクセスするのは困難なので、ノートPCを安全に運用できる。メールに添付されて侵入してくるウイルスなどに対処するのは困難だが、これはノートPC側のメールソフトの運用で対処できる。むしろ、一般的なWindows PCで問題になるセキュリティ上の弱点は、ポートの制御などの侵入対策をユーザーが任意に設定することが困難だという点にある。OpenBlockSはこの弱点をうまくカバーできるユニークな解となりうるデバイスだ。

上級者向けのお楽しみも

さて、OpenBlockSはLinuxマシンでもあるので、ルータとして活用するだけでは少々もったいないのは事実である。そこで、何らかのサーバとして利用したくなってくる。しかし、実際問題としてはOpenBlockSの機能をユーザーが拡張するのはかなりLinuxに慣れた上級者でないと困難だ。

最新のファームウェアではテキストエディタが提供されるようになったので、ちょっとした設定変更等は気軽にできるようになり、それほど高度な技術力がなくてもシェルベースで環境設定を変更したりすることは可能になった。その点で、初心者にもちょっと優しくなりつつあるようだ。とはいえ、標準で提供されていないコマンドやサーバ機能を使うとなると、やはり敷居は低くはない。

OpenBlockSはCPUにPowerPCを採用しているため、x86系CPUで利用するバイナリをそのまま動作させることはできない。そこで、PowerPC用のLinuxディストリビューションであるLinuxPPCのパッケージをいくつか入手してOpenBlockSにインストールを試みたが、ダイナミックリンクライブラリがないなどの理由で動作させることはできなかった。出来合いのバイナリパッケージをインストールするだけで動かす、というのはまず不可能だと思ったほうがよいだろう。そのため、OpenBlockS用のソフトウェア開発環境を構築してソースからコンパイルしてやる必要があるが、これにはOpenBlockSにハードディスクを追加することが必須となる。

ぷらっとホームのWebサイトでは、最低限の開発環境を構成したルートパーティションのイメージファイルが提供されているので、これを利用すれば少なくともgccは使える環境がすぐに実現する。しかし、実際にはMakeの際にソースを変更してやる必要も出てくると予想されるため、gccさえあればあとはMake一発で、というわけにはやはりいかないだろう。ハードディスクを導入してしまうと手軽さが損なわれるように思うため、こうした「重装備サーバ」に仕立てるのは少々もったいないように思う。しかし、機能満載で至れり尽くせりの最近のLinuxディストリビューションに飽き足らないユーザーは、まだ未開拓のまっさらの状態に近いOpenBlockSに自分好みの環境を構築することに挑戦してみてもまた楽しめるのではないだろうか。

●

OpenBlockSは、その筐体のコンパクトさに最大の価値がある。常時携帯しての運用や、通常のPCを設置しにくい場所での運用など、状況に応じていろいろな利用が考えられる。Linuxマシンとして見た場合には、一般的なディストリビューションとは環境が違っているため、不慣れなユーザーは面食らうかもしれない。しかし、基本を理解していれば自力で楽しめる余地が大きいだけに興味深いデバイスといえるだろう。



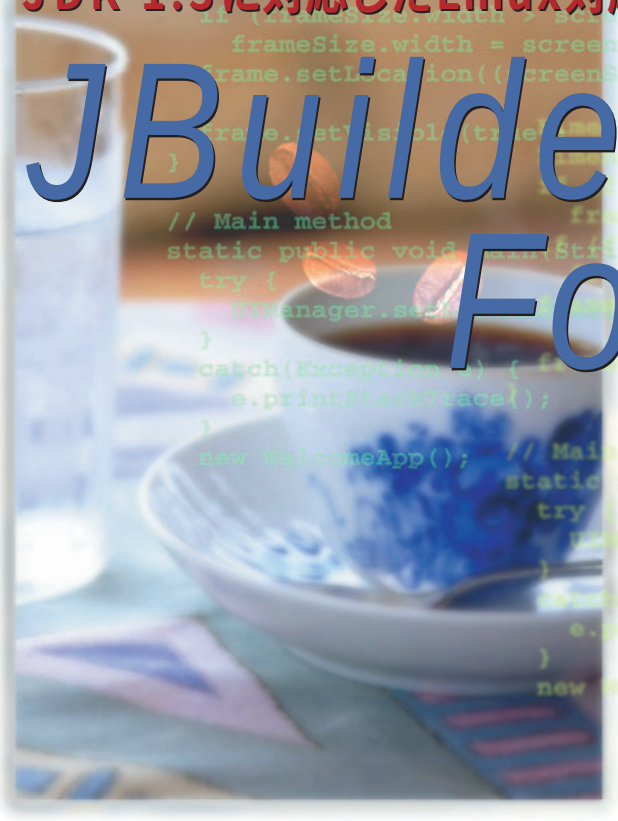
写真3
拡張インターフェイス基板はCompact Flashとハードディスクの両方が載る。

製品名	OpenBlockS
CPU	PowerPC 860T 50MHz
標準Flash ROM	4Mバイト
標準RAM	16Mバイト
シリアルポート	RS-232C×1ポート
イーサネット	10/100BASE-T×1ポート 10BASE-T×1ポート
拡張インターフェイス	IDEインターフェイス×1 Compact Flashカードインターフェイス×1
OS	Linux 2.2.13
オプション	2.5インチハードディスク Compact Flashメモリ
外形寸法	85(W)×120(D)×50(H)
最大消費電力	2.5A（DC 5V、起動時）

短期集中連載

JDK 1.3に対応したLinux対応のJavaビジュアル開発ツール

JBuilder 4 Foundation



第2回　JDKを入れ替えて使ってみよう

文：加藤大受
Text：Daiju Kato　daiju@ms.tokyo.jcom.ne.jp

新しいJBuilder 4 Foundationは、異なるJDKを使用する機能を持っています。今回は、JDKを入れ替えて、新しいJDKを使ってみましょう。また、レイアウトマネージャについて説明し、簡単なプログラムを作成してみます。

JBuilder 4 Foundation for Linuxには、IBMのJDK 1.3 Standard Edition（以下、JDK 1.3SE）が付属しています。IBMは、Javaの本家のSunよりも先にJDK 1.3SEをリリースしました。IBMはJDK 1.1.8から、JDK 1.2を飛ばしてJDK 1.3の採用を決めたわけです。JBuilder 4にIBMのJDKが搭載されているのも、最初にリリースされたものだからといえるでしょう。

統合開発環境（IDE）には、1つだけJDKを登録することができ、デフォルトがIBMのものとなっていますが、今回のJBuilder 4 Foundationでは、異なるJDKを使用することが可能となっています。ここでは、Sunの最新のJDK 1.3.0_01を設定してみましょう。

まず、SunのWebサイトからLinux版のJDK 1.3.0_01をダウンロードします。RPMとtar.gz版の2種類が用意されており、どちらも自己解凍型で、シェルスクリプトを実行するとrpmやtar.gzファイルを展開します。**画面1**では、ダウンロードしたRPM形式のものを展開してインストールしています。インストール先は/usr/java/jdk1.3.0_01となります。続いて、JBuilder 4 Foundationを起動して新しくインストールしたJDKを設定しましょう。

メニューから、［ツール］－［JDKの設定］を選択すると**画面2**のような画面が表示されます。ここで［JDKへのパス］のボタンをクリックして新しくインストールした、SunのJDKがインストールされているディレクトリを選択します（**画面3**）。もし、IBMのものに戻したい場合は、同じ設定を行うことでIBMのJDKを使用することができます。

ビジュアルなデバック機能

JBuilder 4 Foundationにはビジュアルなデバック機能（**画面4**）が提供されています。デバックはプログラミングを行ううえで必要不可欠で、デバックのやり方に慣れてくればくるほど、短時間で自分の書いたコードの不具合の部分を見つけることができ、修正していくことができるようになります。簡単にまとめると、JBuilder 4 Foundationのデバッガは以下の機能を持っています。

- **変数の値を表示**
- **ソースにブレークポイントを設定（行指定、条件指定）**
- **エラー内容の表示（論理エラー、実行時エラー）**
- **ソースコードレベルでのトレース実行、ステップ実行**

プログラムの実行中に実行時エラー

http://java.sun.com/j2se/1.3.0_01/ja/

表1　SunのLinux版JDK 1.3.0_01ダウンロードサイト

が発生した場合、メッセージペインにエラーが表示されます。Java言語の場合、エラーが発生した行が表示されるため、エラーが発生した行の前からトレース実行、またはステップ実行を行っていくことで、問題の部分をすぐに見つけだすことができるでしょう。論理エラーの場合は、問題のありそうな部分にブレークポイントを設定し、**画面5**のようなデータ監視式ビューに監視したい変数を設定することで、変数に格納されている値などを参照することができます。

JDKで行うデバックに比べ、JBuilder 4 Foundationでのデバックは、ビジュアルなユーザーインターフェイスを持つので初心者にも扱いやすく、またある程度プログラミングに慣れた人にでも使いやすいように設計されています。

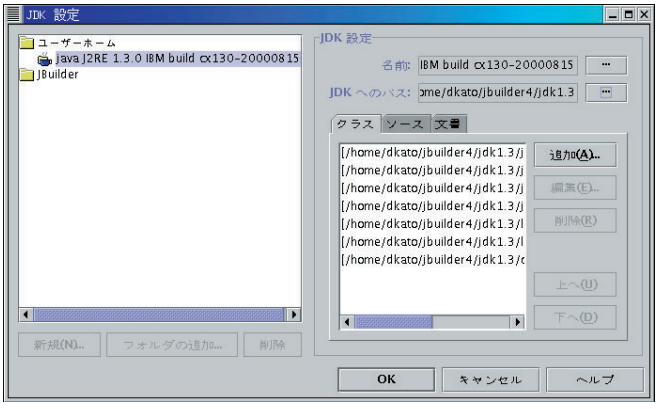
```
# sh j2sdk-1_3_0_01-linux-rpm.bin
                JavaTM 2 SDK v1.3.0_01 Binary Code License Agreement

PLEASE READ THE TERMS OF THIS AGREEMENT AND ANY PROVIDED SUPPLEMENTAL LICENSE
TERMS (COLLECTIVELY "AGREEMENT") CAREFULLY BEFORE OPENING THE SOFTWARE MEDIA
PACKAGE. BY OPENING THE SOFTWARE MEDIA PACKAGE, YOU AGREE TO ALL OF THE TERMS OF
                :
                :
Do you agree to the above license terms? [yes or no]
yes
Unpacking...
Checksumming...
0
0
Extracting...
UnZipSFX 5.40 of 28 November 1998, by Info-ZIP (Zip-Bugs@lists.wku.edu).
  inflating: j2sdk-1_3_0_01-linux.rpm
Done.

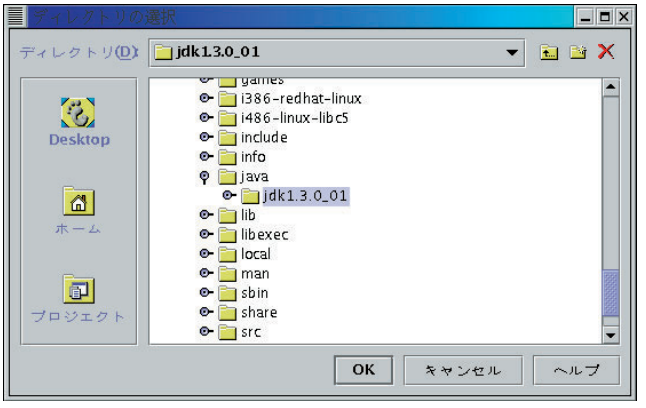
# rpm -ivh j2sdk-1_3_0_01-linux.rpm
# /usr/java/jdk1.3.0_01/bin/java -version
java version "1.3.0_01"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.3.0_01)
Java HotSpot(TM) Client VM (build 1.3.0_01, mixed mode)
```

ライセンス規約の表示（スペースで次ページへ）
ライセンス規約に同意するかの確認が表示される
「yes」と入力
j2sdk-1_3_0_01-linux.rpmが作成される
作成されたRPMファイルのインストール
インストールされたJDKのバージョンの確認

画面1　SunのJDKのインストール



画面2　JDKの設定のダイアログボックス



画面3　新しいJDKの選択

デバッカの機能をうまく利用することで問題をすぐに修正していくことができるでしょう。

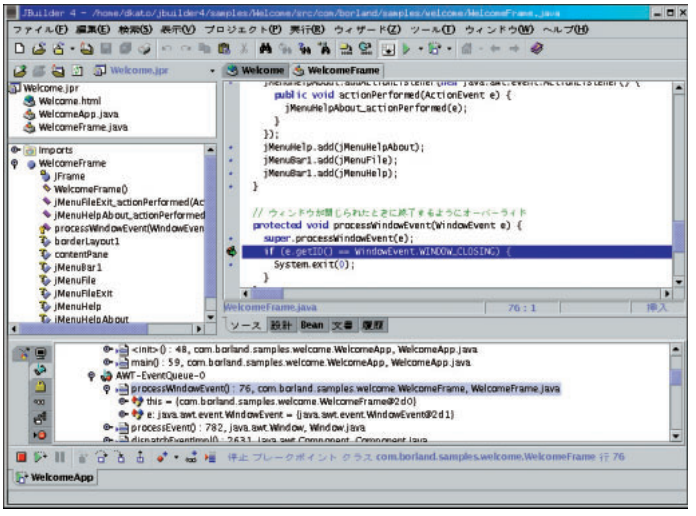
Javaのユーザーインターフェイスを構成する

Javaでユーザーインターフェイスを作成する場合、AWT（Abstract Window Toolkit。**画面6**）または、JFC/Swingと呼ばれるパッケージ（**画面7**）を使用します。これらのパッケージが提供するクラスは、大きく分けて次の3つに分類されます。

・UI部品（コンポーネント）
ボタン、メニュー、エディットボックスなどの個々の部品と、それらを配置する入れ物にあたるパネルやウィンドウのクラス

・配置管理（レイアウト）
コンテナにコンポーネントをどのように配置するかを管理するためのレイアウトクラス

・その他
グラフィックス、イベントやフォント、色といったGUIを表示したり、制御したりするクラス

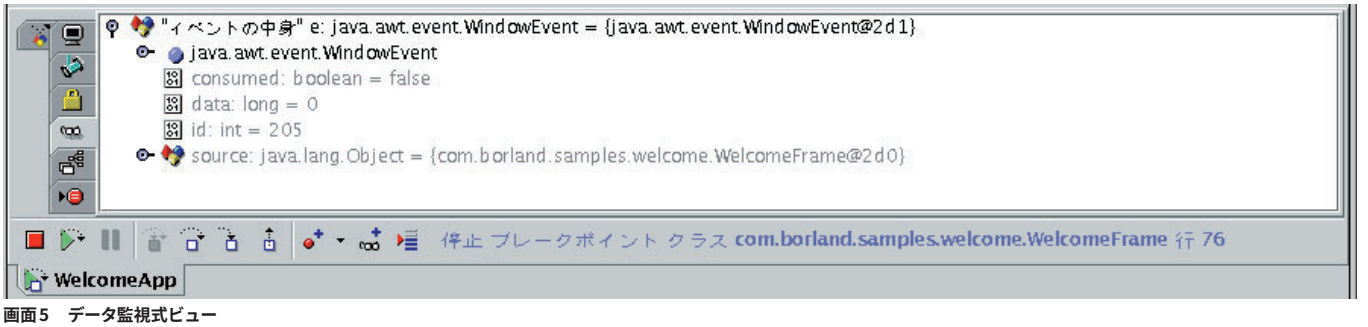


画面4　JBuilderのビジュアルデバッガ

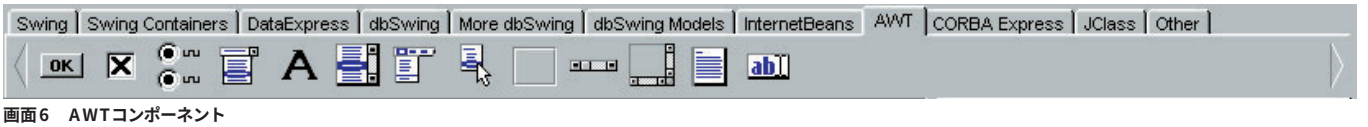
Javaはマルチプラットフォームに対応した言語です。このため、異なるプラットフォーム上でもユーザーインターフェイスなどを綺麗に表示することができるように、レイアウトマネージャという機能があります。GUIを構成するコンポーネントと、このレイアウトマネージャをうまく活用していくと、さまざまなプラットフォームで実行可能なアプリケーションを開発することができるのです。

Javaのユーザーインターフェイスのレイアウトについて

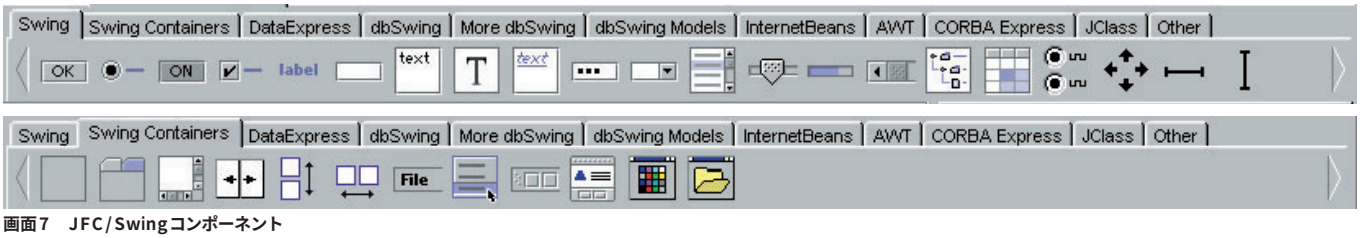
Javaでユーザーインターフェイスを



画面5　データ監視式ビュー



画面6　AWTコンポーネント



画面7　JFC/Swingコンポーネント

作成するとき、必ずレイアウトマネージャを使用します。レイアウトマネージャは、ユーザーインターフェイスを構成するコンポーネントのコンテナの中での大きさや、位置を調整する機能を持っています。たとえば、ウィンドウの大きさが変化したとき、コンポーネントが片寄ったり、大きさのバランスが崩れたりすることなく、レイアウトを綺麗に調整してくれます。

JBuilder 4 Foundationには、JDKで提供されている5つのレイアウトクラスが用意されています。これらのレイアウトクラスは、ユーザーインターフェイスを構成するコンポーネントの配置を決定するだけでなく、PanelやJPanelなどを利用して各レイアウトを組み合わせることで、解像度やプラットフォームに依存しない柔軟なユーザーインターフェイスの作成を可能にします。**表2**は5つの基本的なレイアウトマネージャと、その機能を簡単にまとめたものです。

簡易電卓の作成

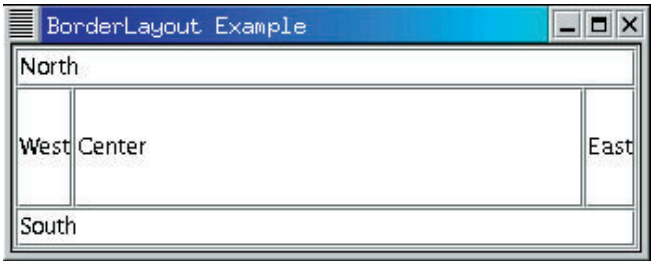
さきほど説明したレイアウトマネージャを使用して、簡単な電卓を作成してみましょう。

JBuilder 4 Foundationを起動し、［ファイル］－［新規プロジェクト］を選択して、プロジェクトウィザードを起動します。プロジェクトウィザードが起動したら、「プロジェクト名」を「Calc」とし、［終了］ボタンをクリックして、プロジェクトファイルを作成します。

続いて、［ファイル］－［新規］で表示される「オブジェクトギャラリ」のダイアログボックスで、「アプリケーション」をダブルクリックし、アプリ

BorderLayout	5つの領域（上部、下部、右、左、中央）に分けてコンポーネントを配置する。上部、下部、右、左、中央は「North」「South」「East」「West」「Center」で設定し、各コンポーネントの「Constraints」プロパティで設定する（**画面8**）
CardLayout	同一の領域に代わる代わるに表示するために使うレイアウトで、ちょうどカードが重なっているかのように配置され、各コンポーネントが何かのアクションによって一番上に表示されるようになる（**画面9**）
FlowLayout	コンポーネントを可能な限り横一行に配置するレイアウトで、入らない場合は複数行になる。コンポーネントの揃え方として、「LEFT」「CENTER」「RIGHT」があり、hgapおよび、vgapプロパティでコンポーネント間の間隔の設定が可能（**画面10**）
GridLayout	コンポーネントを格子状に縦横に並べて配置する。配置する行数と列数の指定が可能。コンポーネントの大きさは一番大きいものに合わせられる（**画面11**）
GridBagLayout	GridBagLayoutは、GirdLayoutを拡張したもので、基本的にコンポーネントを格子状に配置するが、行や列を飛ばしたり、複数行や複数列にまたがったりと、柔軟に配置が可能なレイアウト。配置の細かい設定は、各コンポーネントのConstraintsプロパティで行う（**画面12**）

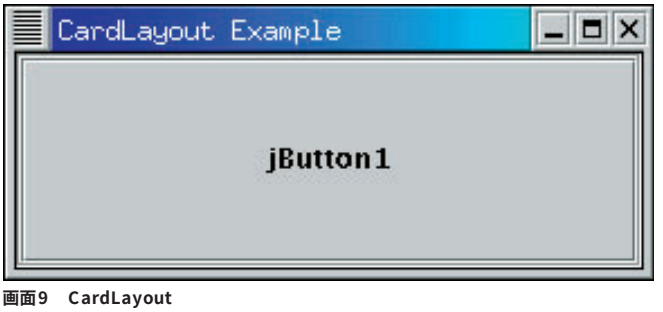
表2　レイアウトマネージャ



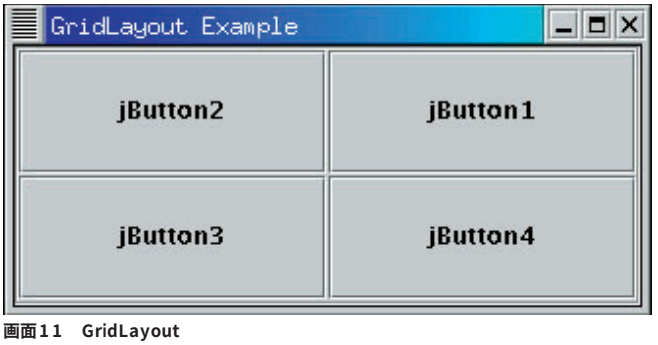
画面8　BorderLayout



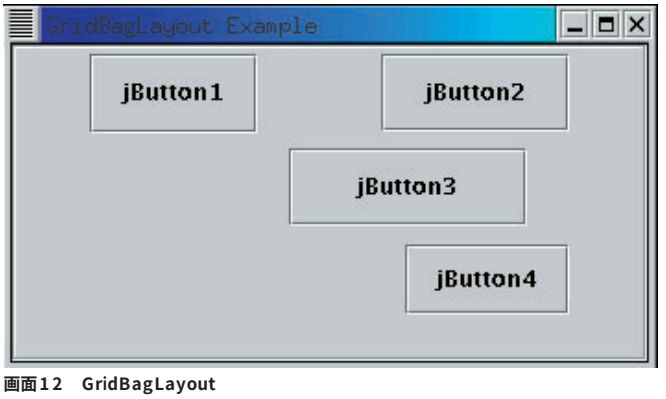
画面10　FlowLayout



画面9　CardLayout



画面11　GridLayout



画面12　GridBagLayout

ケーションウィザードを起動します。アプリケーションウィザードが起動したら、クラス名に「CalcApplication」を設定し、［次へ］ボタンをクリックします。次の画面では、Frameクラスを「CalcFrame」に設定し、フレームタイトルを「簡易電卓」として、［終了］ボタンをクリックします。

次に、CalcFrameクラスがソースコードエディタに表示されていることを確認して、ビジュアルデザイナを起動します。デザイナ画面が表示されたら、「Swing Containers」のページにある、JPanelコンポーネントをフォーム上に2つ配置します。続いて、コンポーネントツリーから「ContentPane」を選択し、インスペクタでlayoutプロパティを「GridBagLayout」に変更します。これで、コンポーネントツリーに「gridBagLayout1」オブジェクトが表示されます。

ここで、レイアウトを整形するために、jPanel1オブジェクトを選択し、Constraintsプロパティ（**画面13**）を**表3**のように設定します。

同様に、jPanel2オブジェクトのConstraintsプロパティを**表4**のように設定します。

jPanel1には数値の表示用に、JTextFieldコンポーネントを配置します。jPanel1のLayoutプロパティをGridLayoutに、JTextFieldコンポーネントのnameプロパティを「jTextField1」から「DisplayField」に変更します。

続いて、jPanel2オブジェクトのLayoutプロパティを「GridLayout」に変更し、コンポーネントツリーからjPanel2の下の「GridLayout2」を選択し、インスペクタでcolumnsプロパティとrowsプロパティを4に設定します。

今度は、電卓のボタン用に16個のJButtonコンポーネントをjPanel2に配置します。配置後、jPanel2のLayoutプロパティを「GridLayout」から「GridBugLayout」に変更し、4行目の右から2番目のボタンを削除し、ボタンの数を15個に変更します。

最後に、**画面14**のように各ボタンのテキストを変更します。ENTERボタンの部分については、ほかのボタンの2つ分の幅を設定しており、Constraintsプロパティの設定は**表5**のようになります。

ユーザーインターフェイスの設計が終わったら、続いてボタンを押したときの処理のコードを書きます。「0」のボタンを選択し、インスペクタからaction_performedのイベントを選択してダブルクリックしてください。コードエディタが表示され、ボタンを押したときに実行されるコードが記述できるようになります。

Javaはイベントドリブンのプログラミング言語です。つまり、何らかのアクションが発生すると、そのアクションに指定されているコードが実行されます。この流れは**図**のようになります。

数値の0を追加すると考えると難しいですが、文字列で処理してしまえば単に末尾に0を足すだけになりますので、0～9までのボタンを押したときは単純に表示領域に押したボタンの文字を追加するだけとなります。たとえば、0のボタンを押したときの処理は**リスト1**のようなコードになります。



画面13　Constraintsプロパティダイアログボックス

グリッド位置	X:0	Y:0	幅:1	高さ:1
外部余白	上:1	左:0	下:1	右:0
内部余白	幅:390	高さ:70		
ウェイト	X:1.0	Y:1.0		
アンカ	C			
拡張	両方			

表3　jPanel1オブジェクトのConstraintsプロパティ

グリッド位置	X:0	Y:2	幅:1	高さ:5
外部余白	上:1	左:1	下:1	右:1
内部余白	幅:390	高さ:56		
ウェイト	X:1.0	Y:1.0		
アンカ	C			
拡張	水平			

表4　jPanel2オブジェクトのConstraintsプロパティ

グリッド位置	X:1	Y:3	幅:2	高さ:1
外部余白	上:0	左:1	下:1	右:1
内部余白	幅:113	高さ:45		
ウェイト	X:0.0	Y:0.0		
アンカ	W			
拡張	なし			

表5　ENTERボタンのConstraintsプロパティ

同じように、1〜9までの処理を追加します。

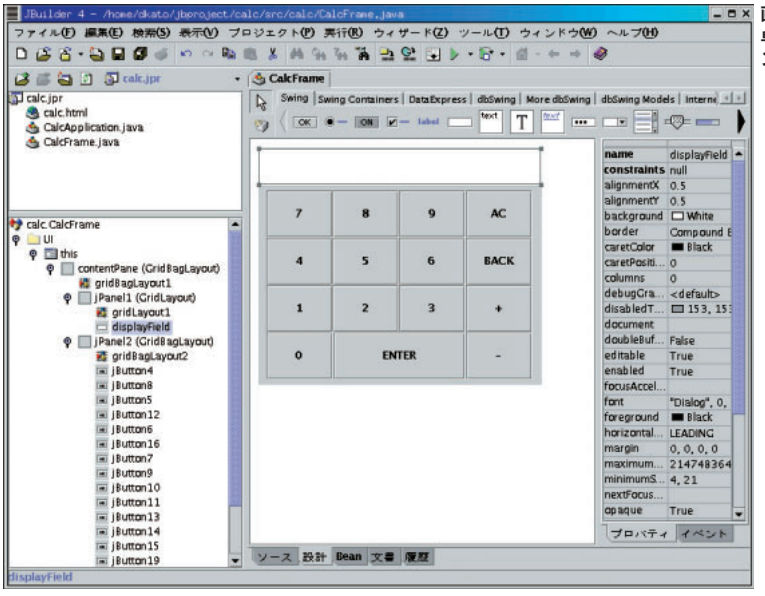
続いて、BACKボタンを押したときの処理を考えます。BACKボタンを押すと、末尾の1文字が消えるようにします。プログラム的に考えると、現在の文字数を取得し、「現在の文字数−1」だけ表示領域に戻すようなコードを書いてやることになります。コードは**リスト2**のようになります。try…catchというフレーズで囲まれていることがわかると思います。これは例外処理で、tryフレーズ内でエラーが発生したとき、catch以下のコードが実行されることになります。末尾の文字を削除すると説明しましたが、実際は先頭から取得する文字数を指定することで処理しています。文字数−1なら最後の文字は取得されないことになるということです。

プラス処理およびマイナス処理ですが、単に文字列の追加や削除だけでは処理できません。これらのボタンを押したときに、今まで表示していた値をどこかに格納してやらなければなりません。表示しているのは単純に文字列なので、いったん文字型の変数に格納してやります。この格納する変数の宣言場所ですが、ボタンを押したときに実行されるイベント内に宣言すると、ほかのイベントから参照することはできません。このため、イベントの外側で宣言し、すべてのイベントから参照できるようにします。宣言の場所はボタンなどのオブジェクトが宣言されているうしろに追加します（**リスト3**）。

ここでは、bufferという文字列変数に格納しています。プラスを押したのかそれともマイナスを押したのかを判別するために、isPlusという論理型の変数を用意しています。この論理型の変数があることで、あとでENTERキーを押したときに、プラスなのかマイナスなのかがわかるのです。

AC処理は単純に表示領域とバッファをクリアしてやるだけ、つまり空白文字列をセットします（**リスト4**）。

最後にENTER処理ですが、isPlusの変数によってプラス処理かマイナス処理かの判別を行い、計算を実行して表示領域に結果を表示します（**リスト5**）。計算は数値型に変換して行う必要



画面14　簡易電卓のユーザーインターフェイス

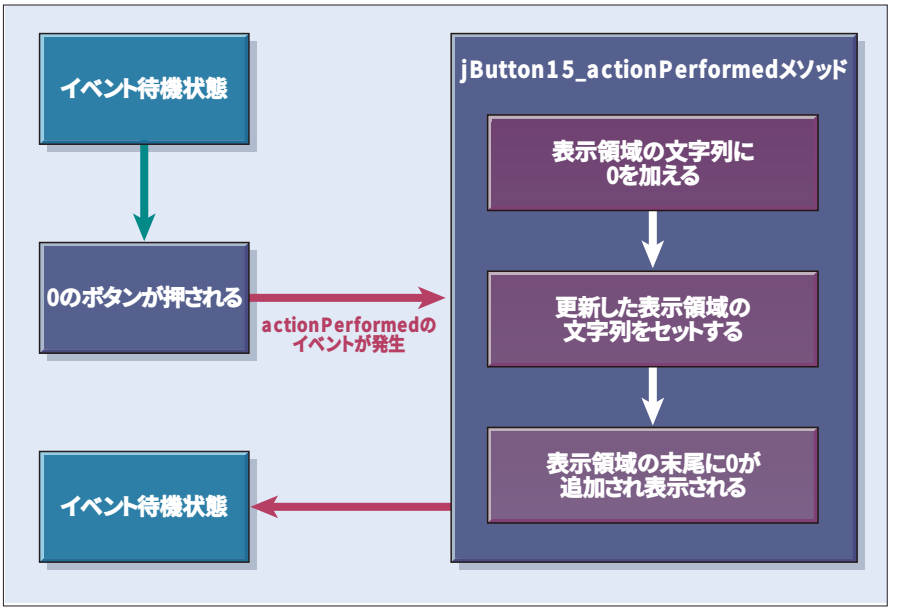


図　0のボタンが押されたときの流れ

▼リスト1　0のボタンが押されたときの処理

```
//0を入力
void jButton15_actionPerformed(ActionEvent e) {
    displayField.setText(displayField.getText()+"0");
}
```

があります。バッファも表示されている値も文字列なので、Integerクラスの ParseIntメソッドを使って数値型に変換し、計算の実行後、文字型に戻して表示領域にセットします。ここではtotalという数値型の変数を宣言し、いったんtotalに計算結果を格納してから、文字列型に変換して表示領域にセットしています。

文字列と数値の変換は次のようなメソッドが用意されています。

文字列を数値へ

```
Integer.ParseInt(<文字列>)
```

数値を文字列へ

```
Integer.toString(<数値型>)
```

コードの作成が終了したら、実行して動作を確認してみましょう（**画面15**）。コード自体はそれほど難しいものではないので、問題なく動作するでしょう。時間のある方は割り算、かけ算を追加してみるといいでしょう。割り算を追加する場合、整数型で終わることは希なので、計算値はInteger型ではなくFloat型となりますので注意してください。また、0で除算した場合のエラー処理も必要となるでしょう。

JBuilder 4 Foundationでは、レイアウトマネージャとビジュアルデザイナを使用していますので、このような複雑なユーザーインターフェイスのようなものでも簡単に設計することができ、本来のコードを書くことだけに専念することができます。今回のサンプルを改良しながらJavaのレイアウトマネージャをマスターしてみるといいでしょう。

▼リスト2　BACKボタンが押されたときの処理

```
//BACK処理
void jButton9_actionPerformed(ActionEvent e) {
     try {
          //表示領域の文字列を取得する
          int i = displayField.getText().length();
          //先頭から文字数-1の文字を取得し、再度セットする
          displayField.setText(displayField.getText(0,i-1));
     } catch (Exception ex) {

          ex.printStackTrace();
    }
}
```

▼リスト3　保存用バッファ、プラスかどうかの論理変数の宣言

```
        :
        :
JButton jButton18 = new JButton();
JButton jButton19 = new JButton();
GridBagLayout gridBagLayout2 = new GridBagLayout();

//入力値の保存用のバッファの宣言
String buffer = new String();
//プラスかどうかの論理変数の宣言
boolean isPlus = true;

/**フレームの構築*/
public CalcFrame() {
        :
        :
```

▼リスト4　ACボタンが押されたときの処理

```
//AC処理
void jButton12_actionPerformed(ActionEvent e) {
    //バッファと表示領域のクリア
    buffer = "";
    displayField.setText("");
}
```

Javaの言語仕様について

さきほど作成した簡易電卓のプログラム内では、ほとんどJavaの言語仕様について解説しませんでした。詳細な言語仕様については市販の書籍にお任せしますが、ここでは初めてプログラミングをされる方を対象に、用語と言語仕様を説明しましょう。

まず、Javaというプログラミング言



画面15　実行したところ

語は、オブジェクト指向と呼ばれる特性を持っています。一般的に、JavaはC++言語の特徴を継承しており、C++言語の持つ難解な部分を削除してわかりやすい言語として開発されたものです。オブジェクト指向は、プログラムコードを構造化することで再利用性と柔軟性を高め、メンテナンスしやすいように考えられたものです。

C++言語、Java言語はクラスという概念を持っており、クラスはそのクラスが属する（メンバと呼びます）関数（一般的には関数ではなくメソッドと呼びます）や、そのクラス内で使用される変数（フィールド変数、プロパティと呼ばれます）からなります。

Java言語の場合、プロパティに値を代入したり、プロパティの値を参照したりすることは禁止されています。値を代入するときは、セッターと呼ばれるメソッド（setText()など）を、値を参照するときはゲッター（getText()など）を利用します。

一般的に、クラスは必ず親クラスを持っています。たとえば、先ほどの簡易電卓のフレームクラスCalcFrameの親クラスはJFrameクラスです。JFrameクラスを継承して、CalcFrameクラスを作っているので、ユーザーインターフェイスを作成することが可能なのです。

JFrameクラスといったクラスはJava APIに用意されている、パッケージと呼ばれるライブラリに含まれています。JFrameクラスはjavax.swingに含まれていますので、CalcFrameクラスでは「import javax.swing.*;」とすることで、Swingのライブラリを使用することを宣言しています。

なお、Javaのプログラムは簡単に説明すると**リスト6**のような構造を持っています。

この構造を頭に入れて、CD-ROM内のCalcFrameクラス（CalcFrame.java）を参照してみてください。各メソッドの宣言の最初にコメントを追加してありますので、どこで何を宣言しているのかが少しずつ理解できると思います。

ボタンなどのオブジェクトや、一般の文字型、数値型の変数は**リスト7**のように定義します。

先ほどの簡易電卓で簡単に説明しましたが、変数は宣言する場所によって使用できる範囲が異なります。簡単にいってしまえば、メソッド内で定義したものはそのメソッドの中だけ、メソッドの外で宣言したものはほかのメソッドから参照可能だということです。この仕組みによって、意図しない変数の書き換えなどが発生しないようになっています。

ここで説明した内容を頭に入れてJBuilderを使っていくと、難しく思えるJava言語も少しずつ理解していけると思います。JBuilderを使いながら、市販のJavaのプログラミングの解説書を読んでいくと、Javaの言語特性と、JBuilderがどのような点を手助けして使いやすくしているのかが理解できると思います。より細かい説明については次回、新しいプログラムを作成していく中で説明します。

▼リスト5　ENTERボタンが押されたときの処理

```
//ENTER 処理
void jButton18_actionPerformed(ActionEvent e) {
    //合計値用の変数を作成
    int total;
    if (isPlus) {
        //プラスの場合
        total = Integer.parseInt(buffer)+
                Integer.parseInt(displayField.getText());
    } else {
        //マイナスの場合
        total = Integer.parseInt(buffer)-
                Integer.parseInt(displayField.getText());
    }
    //計算結果を表示
    displayField.setText(Integer.toString(total));
}
```

▼リスト7　オブジェクトの宣言と変数の宣言

```
オブジェクトの型   作成するオブジェクトの名前   = new   オブジェクト名;
    JButton            jButton1          = new   jButton();

変数の型          変数名   [= 初期値 ];
  int             total    = 0;
```

▼リスト6　Javaプログラムの構造

```
import …<パッケージ名>;
public class <クラス名> extends <親クラス> {
                        オブジェクトの定義
                        フィールド変数の定義
    メソッド1 {
        メソッドの中身
    }
    メソッド2{
        メソッドの中身
    }
            :
            :
}
```

PDAや携帯電話にも対応したWebグループウェア
iOffice2000

最近は一時に比べて「Webグループウェア」が話題に登ることは少なくなったようだ。しかし、これはWebグループウェアが注目を浴びることがなくなったというわけではなく、むしろ、日常的に一般に利用されるごく普通のシステムになってきたということではないだろうか。ネオジャパンのiOffice2000は、そうしたWebグループウェアの中でも安定した支持を得ている製品である。

文：渡邉利和
Text : Toshikazu Watanabe

価格　3万9800円（5ユーザー）～99万8000円（無制限）

問い合わせ先　株式会社ネオジャパン
E-Mail　agency@neo.co.jp
FAX　045-912-5975
http://www.neo.co.jp/

インターネット時代のアプリケーション

インターネットの普及は、コンピュータを取り巻く環境を大きく変えた。もちろん、Linuxの急速な発展もインターネット抜きにはあり得なかったろう。オープンソースによる全世界規模の共同開発の実現はインターネットのもっともめざましい恩恵のひとつだが、コンピュータに利用者として接するユーザーにも大きな変化をもたらしている。挙げればさまざまあるのだが、最近話題になっている変化のひとつにASP（Application Service Provider）の台頭がある。

インターネットへのアクセスラインの整備が進み、iDC（Internet Data Center）などの基幹サービスが充実してくると、ASPは俄然、現実的な存在となってくる。机上のPCに必要となるソフトウェアすべてをインストールして利用していた時代から、必要に応じてサーバのリソースを利用するクライアント／サーバの時代を経て、ついにクライアントにはWebブラウザ以外の何も必要としないASPに辿り着いたのである。

とはいえ、ASPは基本的には企業ユーザー、それもある程度以上規模の大きな組織のためのものと感じてしまい、さほど関心を持たないユーザーも多いのではないだろうか。しかし、これは正しい認識とはいえない。ASP、あるいはASP的なアプリケーション利用の形態は、今やインターネットにアクセスする誰もが利用可能であり、現にあまり意識せずに利用しているユーザーも多いはずだ。

たとえば、Webメールのシステムを考えてみよう。Webブラウザでメールが読めるサービスは今では珍しいものではない。無料でメールアカウントを提供してくれるサービスも多数あるが、こうしたサービスはWebメールの形で利用するのが一般的だ。これも、クライアント側にはWebブラウザ以外必要なく、メールのデータも読み書きのためのアプリケーションも基本的にWebサーバ上に構築されている、という意味では極めてASP的なアプリケーションとなっている。

そして、ASP的なアプリケーションのもうひとつの例として挙げられるのがWebグループウェアだ。個人ユーザー向けに同様の機能を「カレンダー」として提供しているWebサイトもある。これなどは完全にASPそのものだといえるのだが、一般にWebグループウェアと言った場合には、LAN内部などのイントラネット環境で利用されるものをイメージすることが多い。ネオジャパンのiOffice2000も、こうしたWebグループウェアシステムのひとつである。

Webグループウェアの構成

WebグループウェアシステムはWebサーバ上に置かれるアプリケーションである。ただし、通常のHTMLだけでは静的なページしか表現できず、「アプリケーション」に求められる処理が実現できない。そのため、処理はWebサーバから呼び出される外部プログラムが実行し、そのプログラムが動的に生成したHTMLファイルをインターフェイスとして利用する、という実装を採

るのが一般的だ。このとき、この外部プログラムをどのように実装するか、さまざまな手段があり得るが、この手段の選び方によっては対応プラットフォームを制限してしまったり、Webブラウザを選んでしまう結果につながることもある。

Webサーバから外部プログラムを起動する手段としては、CGI（Common Gateway Interface）がもっとも古くから利用されており、よく知られている。しかし、CGIによる外部プログラムの起動はサーバの負荷が高くなるきらいがあり、アクセスが集中する大規模サイトなどでは問題となることもある。これに対応して、最近ではWebサーバが独自APIを提供し、これを通じて外部プログラムを起動する手法も広く利用されている。ただし、この方法では外部プログラムの側でWebサーバのAPIに対応する必要があるため、動作プラットフォーム、正確には対応するWebサーバの種類が限定されてしまうという問題がある。

また、表示に関しても標準的な機能だけを使えば問題ないが、マウスの移動に応じ、マウスポインタが特定のアイコンを指したときにはメニューを表示するなどといった処理を行う場合、Webブラウザ上でJavaScriptなどを実行することで実現することになる。このとき、スクリプトの書き方によっては対応するWebブラウザの種類を限定してしまうだろう。

このように、ひと口にWebグループウェアといってもさまざまな構成があり、それが製品の特徴となっていることがわかるだろう。



iOffice2000の特徴

iOffice2000は、標準的な技術を利用することで幅広いプラットフォームをサポートしたWebグループウェアである。OSとしてはSunOS、Solaris、Windows 2000／NT／98／95、Linux、FreeBSD、HP-UX、AIXなどに対応し、WebサーバはSun WebServer、Oracle Web Server、Oracle Application Server、Netscape Enterprise Server、Apacheなどの UNIXプラットフォーム用のものに加え、Windows用のIIS、Peer Web Service、Personal WebServer、Oracle Web Application Server、Netscape Enterprise Server、Apacheなどをサポートしている。データの互換性も確保されており、運用するOSやWebサーバを変更してもデータはそのまま移行できる。

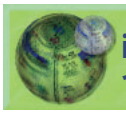
標準では「スケジュール」「仕事リスト」「WebMail」「伝言・所在」「タイムカード」「設備予約」「ワークフロー」「回覧板」「掲示板」「共有アドレス」「電子会議室」「文書管理」「アクションリスト」「プロジェクト管理」「お弁当予約」の15のアプリケーションが利用できる。なかでも、WebMailが含まれているのはちょっと目を引くところだ。ただ、機能の数に目を奪われないよう気をつけたほうがよい。最初に運用を開始する場合にはごく少ない機能から始め、ユーザーが慣れてきたところで徐々に利用する機能を増やすようにしたほうがよいだろう。



画面1　iOfficeの開発・販売元ネオジャパンのホームページ（http://www.neo.co.jp/）



画面2　iOfficeはネオジャパンのホームページからダウンロードして試用できる



iOffice2000のインストールと設定

iOffice2000はさまざまなプラットフォームに対応しているため、インストールに関する注意点もプラットフォームによって変わってくるが、ここではRed Hat Linux 6.2J＋Apacheという環境にインストールしてみた例を紹介する。

インストールしたのは、Ver.2.43の「Linux、FreeBSD版」である。ディストリビューションに応じてさらに数種類が用意されているが、ここではRed Hatに対応したRPMファイルを使用した。RPMファイルは60日間有効の試用版をネオジャパンのホームページからダウンロードできる。そのまま使い続ける場合は、ライセンスキーを購入すればいい（**画面1**、**2**）。

インストール作業は簡単で、入手したRPMファイルをシステムにインストールすれば終わりである。コマンドラインからは、

```
# rpm -i ioffice-2.4-0.i386.rpm
```

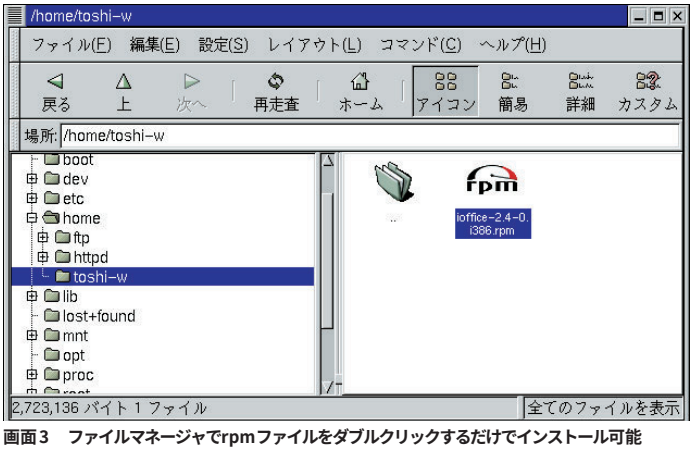
とすればよい。なお、Red Hat Linux 6.2Jで標準となっているGNOME環境では、ファイルマネージャのウィンドウで目的のRPMファイルのアイコンを表示し、ダブルクリックするだけでインストールが完了してしまうので、コマンドラインをタイプする必要すらない（**画面3**）。

インストール先は、実行モジュールが/home/httpd/cgi-bin/ioffice/に、画像データが/home/httpd/html/tmGWImages/にそれぞれ置かれる。このディレクトリは、Apacheのデフォルト設定で参照される場所なので、Apacheを標準設定のまま運用している場合には、iOffice2000のための特別な設定作業は不要で、即利用を開始できる。

まずは、インストールしたサーバ上でWebブラウザを起動し、URLにhttp://localhost/cgi-bin/ioffice/ioffice.cgi? と入力してみよう。**画面4**のような初期画面が表示されればインストールは無事に完了したことになる。念のため、ネットワーク上の別のマシンからも同様にアクセスしてみるとよいだろう。

問題なくアクセスできることが確認できたら、初期設定作業を開始する。まずは、初期画面に表示されている掲示板から「iOffice2000試使用ご利用に挨拶」というリンクをクリックしてみよう。ここに、どう利用を開始し、設定作業をどのように行うかに関する重要な情報が記載されている。この内容に従い、最初はあらかじめ設定されているユーザーのなかから「責任者」を選んで「ログイン」してみる。すると、アプリケーションのアイコンが並んだ画面に進む（**画面5**）。システム全体に関する設定には、「設定」アイコンを利用する（**画面6**）。

ここでは、「マスタメンテナンス」「スタート画面設定メニュー」「メンテナンスメニュー」の3つのカテゴリーに分類されたさまざまな作業項目が一覧表示されている。ここでは、「マスタメンテナンス」の各項目から順次設定していこう。基本的にはフォームに入力していくだけの作業で設定が行える。項目もほぼ見れば意味がわかるものが多く、順番にリンクを辿ってフォームを埋めていくだけで設定は完了するものと思う（**画面7**）。



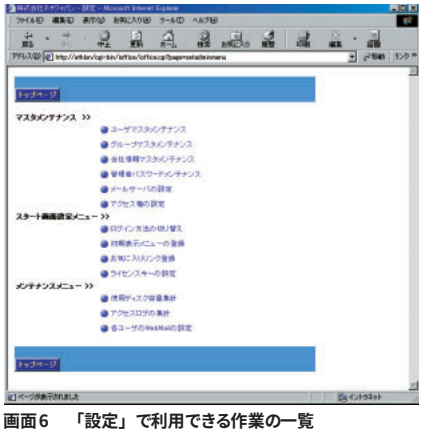
画面3　ファイルマネージャでrpmファイルをダブルクリックするだけでインストール可能

画面4　最初に表示されるログイン画面

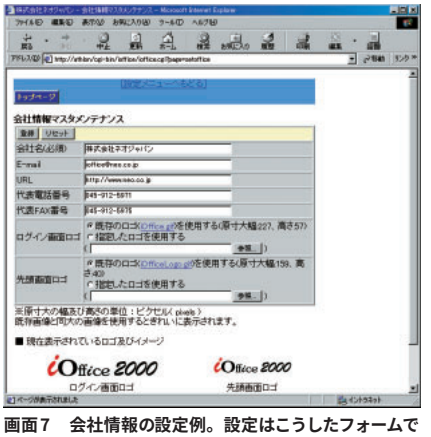




画面5　ログインすると表示されるユーザーごとの初期画面



画面6　「設定」で利用できる作業の一覧



画面7　会社情報の設定例。設定はこうしたフォームで行う

アプリケーションの機能

では、iOffice2000の機能を具体的に見ていこう。

まずは「スケジュール」だ（**画面8**）。ここでは、個人のスケジュールだけでなく、登録ユーザーのスケジュールを確認することができる。スケジュールを設定するには、該当する日の「Add」リンクをクリックすれば入力フォームが表示される（**画面9**）。日付をまたぐ予定なども問題なく入力でき、なかなか使いやすい。スケジュールを公開するか非公開にするかは入力時に設定できるので、個人的なスケジュールを設定して利用してもよいだろう。表示を1日単位に切り替えれば、詳細なタイムバー表示も可能だ（**画面10**）。

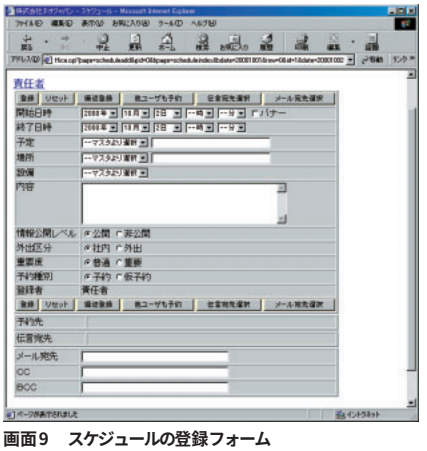
「仕事リスト」は、いわゆるToDoリストのことだ（**画面11**）。全体、あるいは個人ごとに分類項目を設定することなどもできる。他のユーザーのリストを表示することや、他のユーザーのリストを登録することもできるので、管理者が作業を割り振る際に利用してもよい（**画面12**）。

「WebMail」は、Webブラウザ経由で電子メールを利用できる（**画面13**）。利用にあたっては、メールサーバが事前に適切に設定されており、SMTP／POP3を使ってアクセスできる環境が整っていることが前提となる。ディストリビューションや、インストールの際のオプションの指定の仕方によっても異なるが、筆者の実験環境であるRed Hat Linux 6.2Jでは、sendmailはインストールされていたがPOP3はインストールされていなかったため、POP3を利用するためのパッケージのインストールから始めなくてはならなかった。もちろん、iOffice2000と同じサーバ上でメールサーバが動いている必要はなく、リモートのメールサーバでもまったく問題はない。

なお、初めてWebMailを開いた場合には、サーバに接続できずにエラーになるはずだ。左側のリンクから「個人設定」をクリックし、さらに「メールサーバの設定」へと進み、メールを利用するためのユーザーIDとパスワードを設定しよう（**画面14**）。もちろん、これはiOffice2000独自の設定ではなく、メールサーバにアクセスするための情報なので、メールサーバに登録されている情報と同じでなければ意味がない。当然だが、この環境でメールの



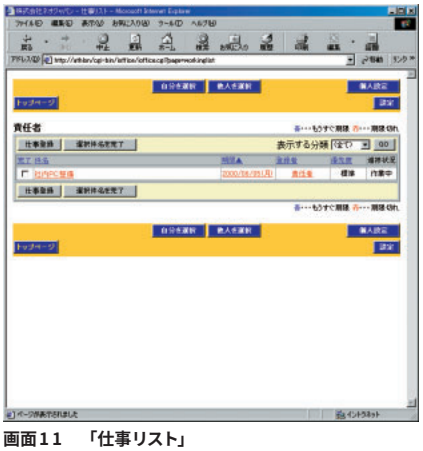
画面8　「スケジュール」



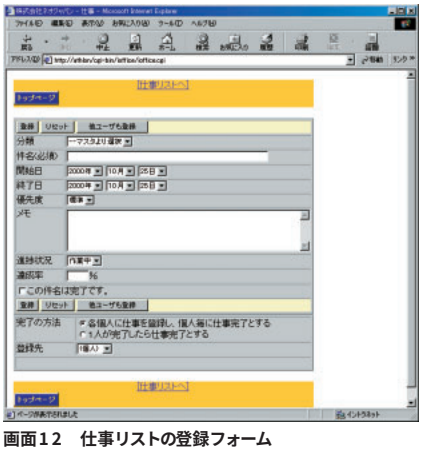
画面9　スケジュールの登録フォーム



画面10　スケジュールの1日表示



画面11　「仕事リスト」



画面12　仕事リストの登録フォーム



画面13　「WebMail」

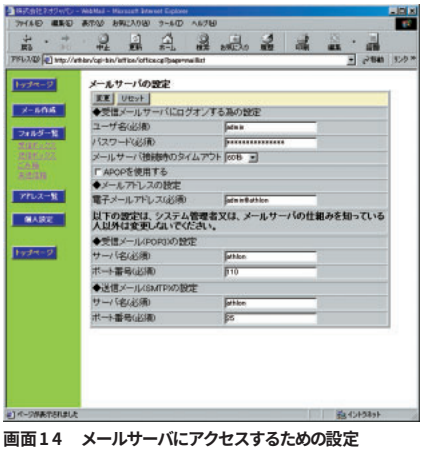
送信も可能で、添付ファイルを付加することなどもできる。メールアドレスに関しても、iOffice2000に登録されたユーザーは自動的にアドレス帳に含まれるほか、共有アドレス帳のデータを参照することができる。

「伝言・所在」は、行き先表示板機能に伝言メモの機能を組み合わせたものだ（**画面15**）。伝言と所在は密接に関連する機能なので、1画面で同時に利用できるようになっているのは便利だ。

「タイムカード」は、見ての通りのものである（**画面16**）。「タイムカードを押す」という操作に相当するのは、ログイン直後のトップページ右上に並んだ「出社」「外出」「戻り」「退社」の4つのボタンをマウスでクリックすることで行う。記録された時刻を手作業で修正することも可能だ。偽造防止策まで期待するのは無理だが、うまく使えば全員にiOffice2000を利用して作業を進めるよう動機づけるための手段としても有効だろう。

「設備予約」も、使い方は簡単に理解できる（**画面17**）。見ての通り、スケジュールのユーザー名を設備名に変更しただけと考えてよい。重複した予約は設定できないようになっているので、単純だがこれで十分実用的だろう。

このほか、多くのアプリケーションが用意されているが、ほとんどは実際に開いてみれば使い方が理解できるものが多い。まずは試してみることをお勧めしたい。



画面14　メールサーバにアクセスするための設定



画面15　「伝言・所在」

Webグループウェアの運用

最後に、管理者として運用上注意すべきことについても簡単に触れておこう。

iOffice2000では、ユーザーごとに管理者権限を付与するかどうかを設定できる。管理者権限があればすべてのデータにアクセスできるため、当然この権限をユーザーにどのように設定するかを考えることは必要だ。とはいえ、これはマルチユーザーシステムに共通の作業であり、改めて指摘するまでもないことだろう。

iOffice2000の機能としては、「設定」の「メンテナンスメニュー」に用意された機能のうち、「使用ディスク容量集計」と「アクセスログの集計」が便利だ。使用ディスク容量の集計は、データの種別ごとにデータ量をKバイト単位で集計し、一覧表示してくれるものだ（**画面18**）。ユーザー個々の使用量も把握できる。Webグループウェアでは、データもアプリケーションもWebサーバ上に置かれることになるため、ディスク容量に気をつけておく必要がある。反面、データが1カ所に集中しているため、バックアップなどのメンテナンス作業は楽になり、信頼性が向上するということも考えられる。

「アクセスログの集計」では、どの機能が多く利用されているのか、あるいは、どのユーザーがどの程度アクセスしているのか、といった情報を得ることができる（**画面19**）。なお、デフォルトではアクセスログは集計しない設定になっているので、運用開始後早いうちにこの設定を変更しておくとよいだろう。

Webグループウェアの運用でもっとも重要な点は、ユーザー全員に利用してもらうように誘導することである。複数ユーザーでの共同作業を支援するために使いやすく工夫された機能がさまざま用意されていることはすでに見てきたとおりだが、実際に活用されな



画面16　「タイムカード」

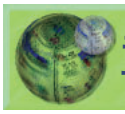


画面17　「設備予約」

くては意味がない。しかも、ほとんどの機能が関係者全員の参加を前提としている点には注意が必要だ。使う人もいれば使わない人もいる、という状況では、個人ベースの単なる便利なツール、ということにとどまってしまう。これでは、「グループウェア」として見た場合の効果は半減どころかほぼゼロに近い。

導入当初は、データの入力が面倒に思えたりすることもあり、なかなか積極的に活用する状況にはならないことが多いだろう。しかし、それを放置してしまうと、導入の効果は得られない。iOffice2000では、「タイムカード」を上手く活用することで、まずは出退社のときには必ずアクセスする、という習慣づけができるはずだ。ここを手がかりに利用を推進していくのがよいと思う。

また、最初からたくさんの機能が用意されているので目移りしがちだが、導入当初のユーザーも不慣れな時期からすべてをいきなり使おうとしても単に情報が分散するだけで混乱を招く可能性がある。最初のうちは利用する機能を限定し、慣れてくるに従って徐々に活用の幅を広げていく、という手段もあるだろう。「設定」－「スタート画面設定メニュー」－「初期表示メニューの登録」では、トップページに表示するアプリケーションアイコンを選択し、使わないものを非表示にすることができる（**画面20**）。



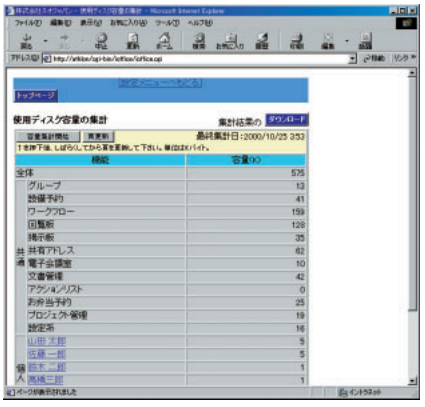
モバイルアクセスのサポート

iOffice2000には、PDA対応版、iモード対応版、EZWeb対応版、Palm-Sync対応版などのバリエーションが用意されている。PDA対応版では、エクシーレ、プラウザ・ボード、WorkPad、Zaurusなど、1／4 VGAサイズのディスプレイを備えたPDAを端末として利用できる。基本的には、表示されるアプリケーション画面を端末の機能に合わせてカスタマイズしてあるもので、PCに比べて小さな画面しか備えていないPDAで効果的に利用できるように配慮されている。PDA対応版で用意されている機能は11種類で、「掲示板」「電子会議室」「文書管理」「アクションリスト」の4つが欠けているという形になる。

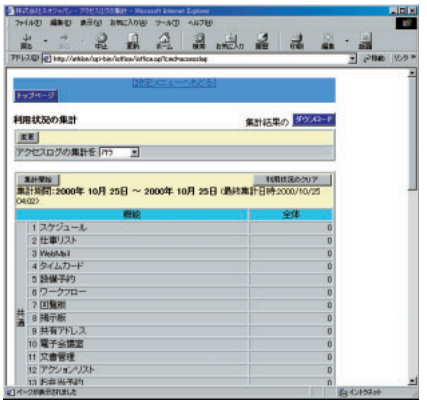
掲示板や電子会議室など、ユーザーからのテキスト入力が多くなるアプリケーションが省かれているのは、やはりPDAの利用環境に配慮したものだろう。利用しにくいアプリケーションが多数用意されていても、カタログを豪華に見せる以上の意味はないからだ。

また、iモード対応版では「スケジュール」「WebMail」「タイムカード」「伝言・所在」「設備予約」「共有アドレス」「仕事リスト」「ワークフロー」「回覧板」の9種類の機能がサポートされている。ほかにも、EZWeb対応版やH"対応版が用意されているので、これらインターネット対応の携帯電話を利用しているユーザーは、外出先などからiOffice2000に蓄積された情報にアクセスすることができる。最近では、重いノートPCを持ち歩くのではなく、PDAや携帯電話／PHS等を活用するユーザーが増えているが、こうしたユーザーに対する配慮もなされている点は、情報システムへのアクセスを社内に限定したくない場合にはありがたい。

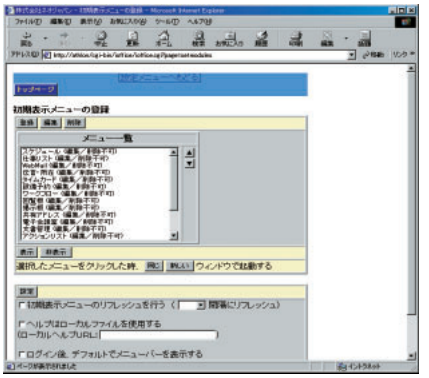
さらに、Palm-Sync対応版では、iOffice2000にアクセスできるPCにPalm DeskTopとiOffice2000 for Palm-Syncクライアントをインストールすることで、このPCにPalmデバイスを接続してデータの同期を行えばiOffice2000上の情報をPalmデバイスに転送できるようになる。セキュリティの問題などで、社外からiOffice2000にアクセスすることを認められない場合でも、Palmデバイスにデータを入れて持ち歩くことができれば便利だろう。社から一歩出ただけですべての情報から遮断されるという情けない思いをせずに済むはずだ。



画面18　「使用ディスク容量集計」



画面19　「アクセスログの集計」



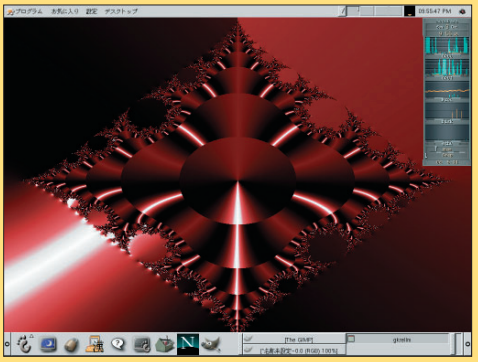
画面20　「初期表示メニューの登録」

Free Application Showcase

文：出井 一
Text : Hajime Dei



FreeAmp P.129



ChBg P.132



Chromium B.S.U. P.140

MP3／OggVorbis／CDとなんでもござれのプレーヤ
FreeAmp 129

Xのルートウィンドウに画像をを表示
ChBg 132

CD-ROMのファイル情報をまとめたカタログを作成
GTKtalog 134

グラフィカルなSamba用ファイルブラウザ
xSMBrowser 136

パーティションの作成やリサイズを行う
GNU Parted / nParted 138

OpenGLを使った美しいシューティングゲーム
Chromium B.S.U. 140

玉を指定された配置に揃えるパズルゲーム
Marbles 141

文字列の送出も可能なソフトキーボード
GTKeyboard 142

自動スクロールするテキストビューア
Reed 143

※紹介したソフトは、すべて付録CD-ROMに収録されています。

MP3／OggVorbis／CDとなんでもござれのプレーヤ

FreeAmp

バージョン：2.1rc5　　ライセンス：GPL

http://www.freeamp.org/

FreeAmpは、LinuxやWindowsなどで動作するフリーな音楽プレーヤソフトだ。MP3とOgg Vorbisに対応し、各種のストリーム再生も可能。さらに、CDDB対応のCDプレーヤとしても使える。最大の特長は、MP3／Oggファイルやプレイリスト、音楽CDの曲目、ストリーム系サイトのブックマークなどを一括管理するMy Music機能だ。もちろん、プレーヤの外見を変えるテーマ（スキン）も用意されている。

●ビルドとインストール

FreeAmpの最新版2.1rc5は、複数のRPMバイナリパッケージで分割配布されている。Red Hat 6.x用なので、たいていのRed Hat系ディストリビューションで動作するはずだ。

まずは、実行に必要なライブラリ（ID3v2とMusicBrainz）のパッケージをインストールしよう。続いて、FreeAmp本体と、サウンド出力用のEsounD／ALSAプラグイン、データベース管理用のCORBAプラグインをインストールする。なお、コマンドライン／ncursesインターフェイスを使わないなら、freeamp-extrasパッケージはインストールしなくてもよい。

ソースからビルドする場合は、FreeAmpのデベロッパーページからスナップショット版のtarボールを入手する。ビルドとインストール手順は、configureスクリプトを用いる一般的なもの。ただし、フリーなアセンブラの「NASM」（http://www.web-sites.co.uk/nasm/）が別途必要だ。

●起動から初期設定まで

FreeAmpを起動するには、ktermなどのコマンドラインから「freeamp&」とする。なお、現在のバージョンでは、以前のようなダイアログの文字化けは起こらない。

ウィンドウはシンプルなパネル風のデザインだ（**画面1**）。再生などのボタンは説明するまでもないだろう。左上の小さなボタン群により、イコライザ表示の追加や、表示サイズの小さなミディアム・ミニビューへの切り替えが可能だ（**画面2**）。

初めて起動した場合は、使用環境に合ったサウンド出力プラグインを選択する必要がある。中央部の[Options]ボタンを押して設定ダイアログを開こう。設定項目はカテゴリごとに分かれているので、左のリストから[Plugins]を選択する（**画面3**）。

変更するのは[Audio Output]の設定だ。EsounD（esd）を使っていれば「esound.pmo」、ALSAなら「alsa.pmo」、どちらも使っていなければ「sound card.pmo」を選択する。「waveout.pmo」を選択して、サウンド出力をWAVEファイル化することも可能だ。

●ファイル名を指定して再生する

まずは、ハードディスク上のMP3／Oggファイルを直接指定して再生してみよう。[Files]ボタンでダイアログを開いて、ファイル名を選択すればいい。ドラッグによる範囲指定や、Ctrl－クリックによる複数選択も可能だ。[OK]ボタンでダイアログを閉じると、すぐに再生が始まる。

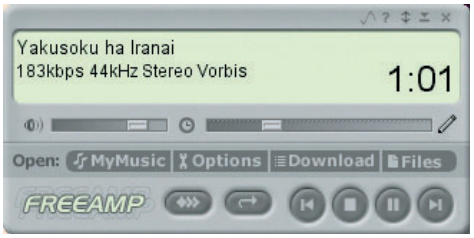
このほか、ドラッグ＆ドロップにも対応しているので、GNOMEのgmcなどからMP3／Oggファイルをドラッグ＆ドロップすることでも再生が可能だ。

もっとも、こうした使い方ではFreeAmpの機能を十分に活かしているとは言いがたい。続いては、FreeAmpの大きな特長であるMy Music機能について説明しよう。

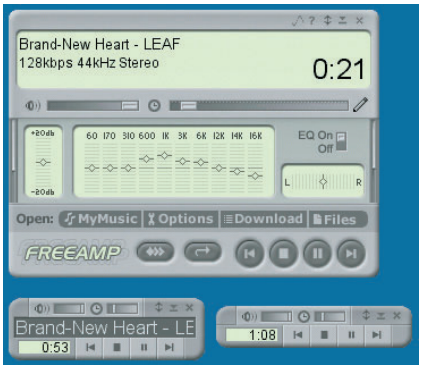
●リソースを統一管理するMy Music

My Music機能は、ローカルなディスク上のMP3／Oggファイルやプレイリスト、インターネットラジオなどストリーム系サイトのブックマーク、CD-ROMドライブに挿入した音楽CDのタイトルや曲目など、FreeAmpで利用する各種リソースをデータベースで一括管理するというものだ。

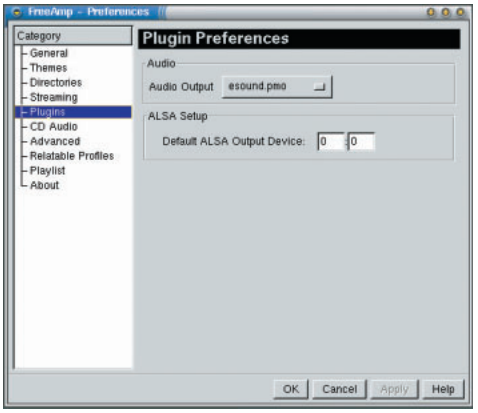
[My Music]ボタンを押してMy



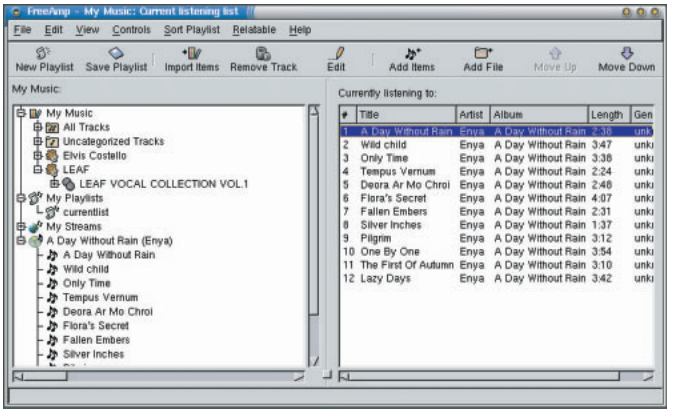
画面1
FreeAmpのウィンドウ。外見がものたりない人はテーマを変えよう。



画面2
イコライザの追加や、表示サイズの変更も可能だ。



画面3
MP3／Oggファイルやプレイリストを My Musicで一括管理。



画面4
サウンド出力プラグインを使用環境に合わせて変更する。

Musicウィンドウを開こう。ウィンドウは2ペイン構成で、左には各種リソースのツリー、右には現在のプレイリストの内容が表示される（**画面4**）。

MP3／Oggファイルやプレイリストを登録するには、ツールバーの[Import Items]ボタンで表示されるダイアログでファイルを選択すればいい。MP3／Oggファイルは[My Music]のツリー、プレイリストは[My Playlists]のツリーに登録される。また、MP3／Oggファイルの場合は、ファイル内のタグ情報に基づいて、さらに[アーティスト名]－[アルバム名]－[曲目]というツリーで分類される。

登録したいファイルが多数あるなら、MP3／Oggファイルとプレイリストを自動検索して登録するウィザード（**画面5**）を利用しよう。My Musicウィンドウを最初に開いた際に表示されるほか、メニューの[File]－[Search Computer for Music]でも同じ機能を呼び出せる。以前のバージョンように途中で固まってしまうこともない。

このほか、聴いている曲の傾向からお勧めのプレイリストやストリーム系サイトを提示するRelatable機能が用意されている（無効にもできる）。

●プレイリストを作成して再生

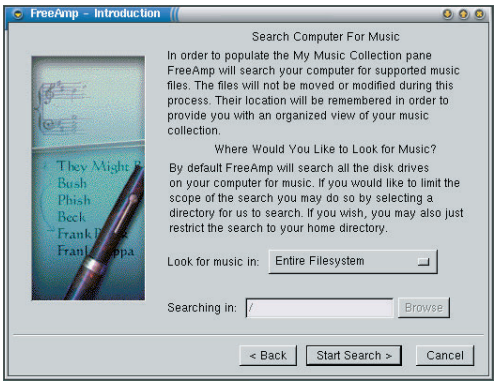
いったんMP3／Oggファイルやプレイリストが My Musicに登録されれば、再生自体はきわめて簡単だ。たとえば、既存のプレイリストやアルバムをそのまま再生するには、ツリー中のプレイリストやアルバムをダブルクリックすればいい。右側のペインに曲目が登録され、再生が始まる。

[My Music]ツリーのMP3／Oggファイルを現在のプレイリストに追加するには、クリックで選択して[Add Items]ボタンを押せばいい。曲順の変更や削除なども可能だ。作成したプレイリストを[Save Playlist]ボタンで名前を付けて保存すると、[My Playlists]ツリーに自動的に追加される。

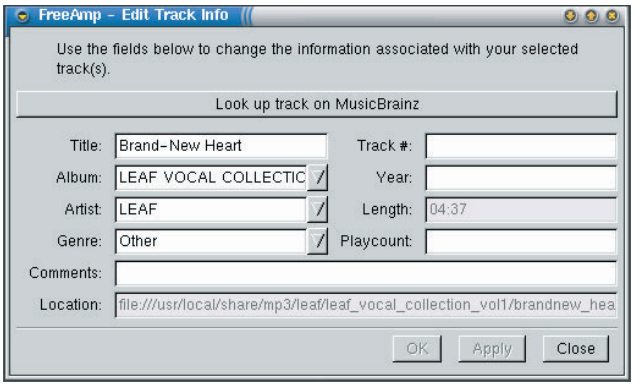
このほか、プレイリストやツリー中のMP3／Oggファイルを選択して[Edit]ボタンを押すと、タグ情報を表示・変更するダイアログが開く（**画面6**）。なお、曲目などに日本語を入力すると、ツリーでは正常に表示されるものの、英語フォント固定のプレイリストやプレーヤの表示は文字化けしてしまうので注意されたい。

●ストリームや音楽CDの再生

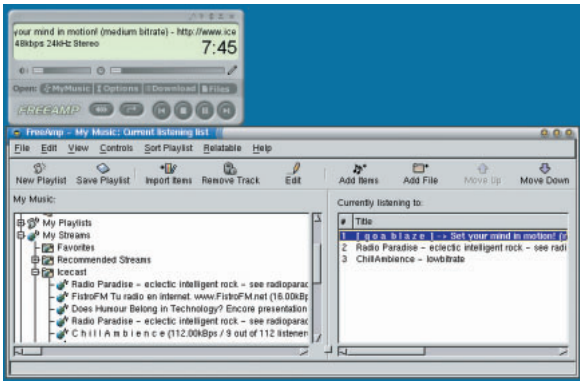
FreeAmpで再生できるのは、ローカルなMP3／Oggファイルにとどまらない。[My Streams]ツリーには、SHOUTcastやIceCastなどを利用したストリーム系サイトのブックマークが登録されており、ローカルなファイル



画面5
ディスク上からMP3／Oggファイルを検索して自動登録できる。



画面6
タグの内容（曲目やアーティスト名など）も変更可能だ。



画面7
通常のファイルと同様の操作でストリーム再生も行える。



画面8
テーマを導入することでプレーヤの外見は大きく変化する。

と同様の操作でプレイリストに登録してストリーム再生が可能だ（**画面7**）。

サイト選択の際は、それぞれのブックマークの末尾に表示されているデータ転送速度と現在のユーザー数を参考にしよう。ISDN/モデムによる接続でも、48kbps程度の速度なら十分実用になる。

また、CD-ROMドライブに音楽CDを挿入し、[CD Audio]ツリーを右クリックして[Update CD info]を選択すると、インターネット上のCDDBサイトから、その音楽CDのアルバム名や曲目情報を取得する。

情報が登録されていれば、[CD Audio]がアルバム名に変化し、その下に曲目のツリーが表示される（見つからない場合は[Unknown Album]、[CD Track 1]などと表示）。MP3／Oggファイルと同様の操作でプレイリストに曲目を登録して再生しよう。

●テーマを導入して外見を変える

FreeAmp標準のウィンドウのデザインは、誰にでも機能がわかるように作られているため、いまひとつ面白みに欠ける。個性的なプレーヤが欲しい人はテーマ（スキン）を使ってウィンドウの外見を変えよう（**画面8**）。

FreeAmpのWebサイトの「Themes!」ページには、30個近くのテーマが用意されている。また、米コカ・コーラ社のWebサイトにも数個のクールなテーマが登録されている。

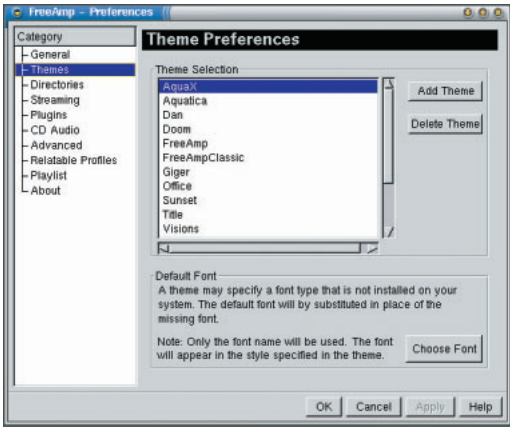
取得したテーマファイル（拡張子.fat）は、/usr/share/freeamp/themesにコピーすれば、FreeAmpが自動的に登録してくれる。その他のディレクトリにある場合は、[Add Theme]ボタンを押して手動登録しよう。

テーマの切り替えは、設定ダイアログの[Themes]ページで行う（**画面9**）。リスト中から切り替えたいテーマを選択して、[OK]（あるいは[Apply]）ボタンを押せばいい。

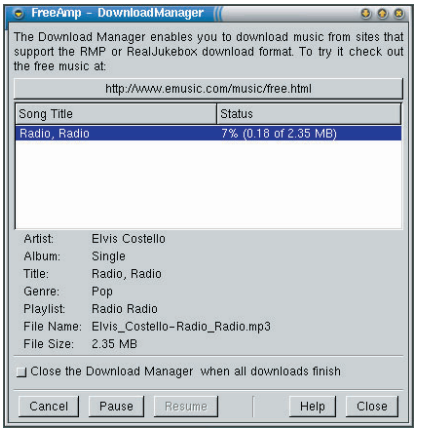
●Webからダウンロードする

FreeAmpは、Real Jukeboxのダウンロード用形式「RMP」（Real Music Package）に対応しており、合法的な音楽サイト、たとえば「emusic」（http://www.emusic.com/music/free.html）などからフリーなMP3ファイルをダウンロードできる。

こうしたページをNetscapeで閲覧し、Real Jukebox用のリンクをクリックすると、FreeAmpのダウンロードブラウザが自動的に起動してMP3ファイルがダウンロードされる（**画面10**）。ダウンロード中は曲の情報が表示され、一時停止や再開も可能だ。ダウンロード後のMP3ファイルは、自動的に[My Music]ツリーに登録される。



画面9
テーマの切り替えは、設定ダイアログの[Themes]ページで行う。



画面10
WebサイトからRMPを利用してMP3ファイルをダウンロード。

Xのルートウィンドウに画像を表示

ChBg

バージョン：1.2　　ライセンス：GPL

http://chbg.sourceforge.net/

ChBgは、Windowsにおける「壁紙チェンジャ」のようなソフトで、登録した画像をXの背景（ルートウィンドウ）に一定時間おきに表示する。imlibやgdk_pixbuf、imlib2などのイメージライブラリに対応しており、ほとんどの形式の画像を表示できる。また、フラクタルなどの計算に基づく画像をシェーディングして表示することも可能。実行にはGTK＋といずれかのイメージライブラリが必要だ。

●ビルドとインストール

ChBgは、tarボールとRPMソース／バイナリパッケージの両方が配布されているので、ディストリビューションに応じて選択しよう。

tarボールからのインストールは、「./configure」→「make」→「make isntall」という一般的な手順だ。利用するイメージライブラリはconfigureスクリプトが自動選択するが、--with-image-libraryオプションで指定することもできる。

RPMのバイナリパッケージは、imlibを使用するものと、gdk_pixbufを使用するものがそれぞれ配布されている。ソースパッケージをリビルドする場合はconfigureスクリプトが自動選択する。なお、どちらも実行ファイルのインストール先が/usr/local/binになるので気をつけたい。

●セットアップウィンドウを開く

「chbg&」とするか、GNOMEメニューの[プログラム]－[グラフィック]－[ChBg]を選択すると、ChBgのセットアップウィンドウが表示される。ジャンル別に設定項目がページ分けされており、上部のタブで切り替えられる。

付属の日本語カタログにより、内容はすべて日本語表示だ。ただし、日本語表示では、一部のタブが隠れてしまうので、少し横幅を広げたほうがいいかもしれない（**画面1**）。

最初に、[セットアップ]ページで表示モードを設定しよう。ChBgは、ルートウィンドウに画像を表示するだけでなく、通常のウィンドウに表示したり、スクリーンセーバとして実行することもできる。

●画像ファイルを登録する

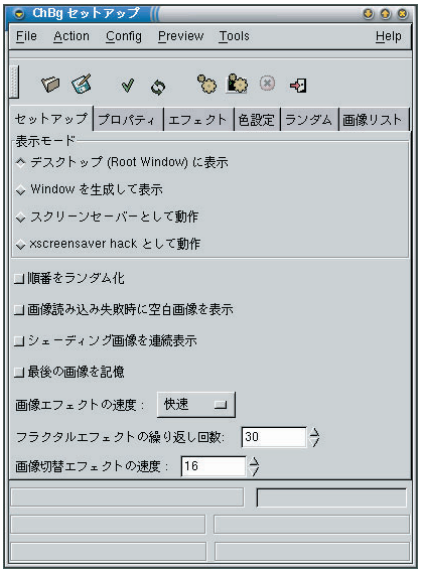
それでは、複数の画像ファイルを切り替えて表示する方法を説明しよう。[画像リスト]ページに切り替え、[画像追加]ボタンを押してダイアログを開く（**画面2**）。

このダイアログでは、ChBgで表示させたい画像を、縮小イメージを見ながら選択できる。ファイル名をクリックするだけで複数選択が可能だ。また、[自動表示]をチェックしておけば、いちいち[表示]ボタンを押さないでも、選択した時点で縮小イメージが表示される。最後に[追加]ボタンを押して、選択したファイルを背景画像リストにまとめて登録しよう。

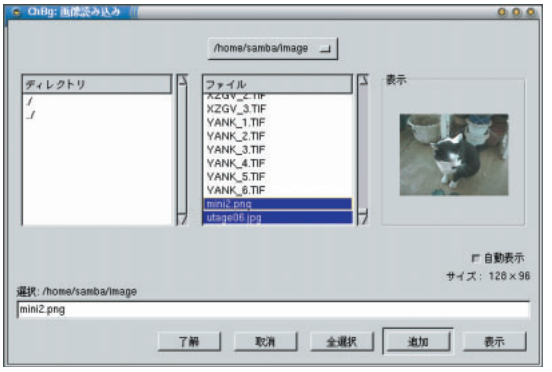
指定したディレクトリ以下の画像ファイルを一括登録することも可能だ。このとき、ファイル名のパターン（拡張子など）を指定したり、小さな画像を除外するためにファイルサイズを制限することもできる。

リストに登録された画像ファイルに対しては、右クリックメニューで選択や削除、画像の参照（ウィンドウ表示）、全画面表示、サムネイル表示などの操作を行える。

特に、独立したウィンドウに表示されるサムネイル（**画面3**）からは、画像の表示はもちろん、スライドショーやルートウィンドウへの表示も行える。つまり、ChBgを一種の画像ビューアとして利用できるわけだ。



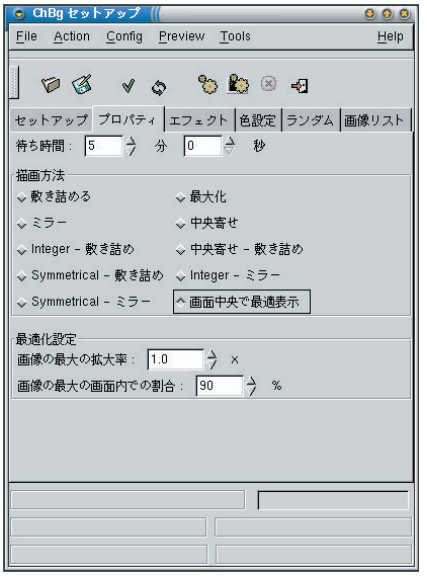
画面1
ChBgの各種設定を行うセットアップウィンドウ。



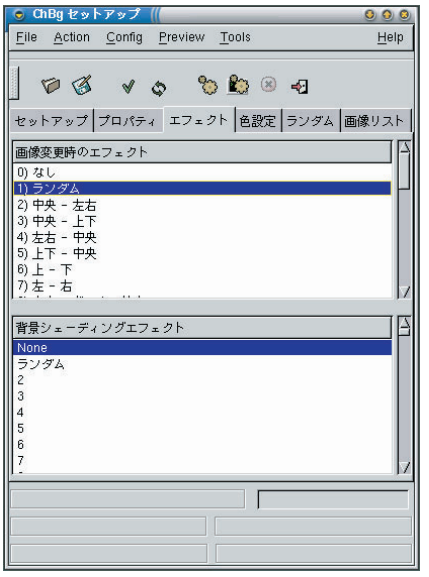
画面2
このダイアログで画像を追加する。プレビューも可能だ。



画面3
登録画像を独立したウィンドウにサムネイル表示する。



画面4
[プロパティ]ページで切り替えの間隔と描画方法を設定。



画面5
画像変更時やシェーディングのエフェクトを選択。

●画像ファイルを切り替えて表示

登録した画像ファイルを一定時間ごとに切り替えて表示するには、[プロパティ]ページで、切り替え後の待ち時間（初期値5分）と、画像の描画方法を設定する（**画面4**）。

描画方法は、「敷き詰める」「最大化」「中央寄せ」などの基本的なものから、画像を反転させながら敷き詰める「ミラー」や、画面にきっちり収まるように縮小して敷き詰める「Integer-敷き詰め」などの凝ったものまで揃っている。

設定が完了したら、ツールバー中央の[現在の設定で実行]ボタンを押して、実際の表示を確認しよう。待ち時間が経過すると次の画像に切り替わるはずだ。なお、待機中に変更した設定は[setupの変更を適用]ボタンを押すか、[画像の更新を停止]ボタンで更新停止するまで有効にならない。

[エフェクト]ページの[画像変更時のエフェクト]では、切り替わる際の画面効果を設定できる（**画面5**）。30種類近くの切り替えエフェクトが用意されているので、「ランダム」に設定してそれらを楽しむのもよいだろう。なお、エフェクトの実行に要する時間は待ち時間には含まれない。

しばらく動かしてみて、好みの設定が固まったら、[Setup windowを隠し、現在の設定で実行]ボタンを押す。これで、セットアップウィンドウが消えて、バックグラウンドで画像の切り替えが行われる。

●シェーディング画像の表示

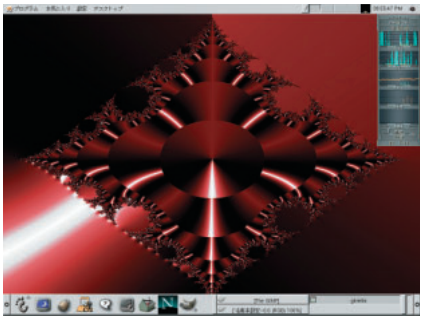
ChBgのもうひとつの楽しみ方は、画像ファイルの代わりに、フラクタルなどの計算に基づくシェーディング画像を表示することだ。

ChBgには、あらかじめ500種類以上のシェーディングエフェクトが用意されており、さまざまな画像を切り替えて楽しめる。

シェーディング画像を表示するには、まず[セットアップ]ページで[シェーディング画像を連続表示]をチェックする。続いて、[エフェクト]ページでは、[画像変更時のエフェクト]を「なし」、[背景シェーディングエフェクト]を「ランダム」に設定する。さらに、[色設定]ページで[シェーディング色モード]を「Gradient」に設定すれば準備完了だ。

[現在の設定で実行]ボタンを押すと、シェーディング画像が生成され（種類によって生成時間は異なる）、ルートウィンドウに表示される（**画面6**）。待ち時間の設定などは画像ファイルの場合と同じだ。

なお、シェーディング画像は画像ファイルの背景として使われるので、シェーディング画像だけを楽しみたいなら、[画像リスト]ページで選択画像のリストを空にしておこう。画像ファイルと組み合わせる場合は、画像ファイルの描画方法を「中央寄せ」など背景が隠れないものにする。



画面6
フラクタルなど計算に基づいたシェーディング画像が表示される。

CD-ROMのファイル情報をまとめたカタログを作成

GTKtalog

バージョン：0.7.0　　ライセンス：GPL

http://gtktalog.sourceforge.net/

GTKtalogは、CD-ROMのファイル情報をデータベース化したディスクカタログを作成するソフトだ。複数のCD-ROMのファイル情報をひとつのディスクカタログでまとめて扱える。作成後は、実際にCD-ROMをマウントすることなく、高速にファイルを検索可能だ。さらに、仮想ファイルシステム（VFS）機能により、tarボールやRPMパッケージ内のファイル情報も取り込める。動作にはGTK＋が必要だ。

●ビルドとインストール

GTKtalogは、ファイル一式をtar＋gzipしたtarボールのみ配布されている。ビルドとインストールは「./configure」→「make」→「make install」という一般的な手順だ。

tarボールにはSPECファイル（gtktalog.spec）が含まれているので、「rpm -ta gtktalog-0.6.3.tar.gz」としてRPMパッケージを作成できる。ただし、RPMのバージョンが3.0.5より古いと、「%makeinstall」マクロが含まれていないため、パッケージの作成に失敗してしまう。

この場合は、tarボールを展開して、SPECファイルの54行目を「%make install」から「make prefix=$RPM_BUILD_ROOT%{prefix} install」に書き換えよう。修正したSPECファイルは/usr/src/redhat/SPECS、tarボールは/usr/src/redhat/SOURCESにコピーし、「rpm -ba /usr/src/redhat/SPECS/gtktalog.spec」とすると、RPMパッケージを作成できる。

●起動と初期設定

「gtktalog&」として起動すると、左右にペイン（領域）を持つウィンドウが開く。左はCD-ROMごとのディレクトリツリー、右は選択したディレクトリ内のファイル一覧という構成だ（**画面1**）。日本語環境では、付属の日本語カタログを利用したメニューやダイアログが表示される。

初めて起動した際には、初期設定を行うため、[編集]－[設定]で設定ダイアログを開こう。まずは、[ディスクセットアップ]ページで、マウントポイント（初期値は/mnt/cdrom）などを確認する（**画面2**）。

続いて、[VFS]ページに切り替え、tarボールやRPMパッケージ内のファイル情報を取り込む「仮想ファイルシステム」（VFS）の設定を確認する。

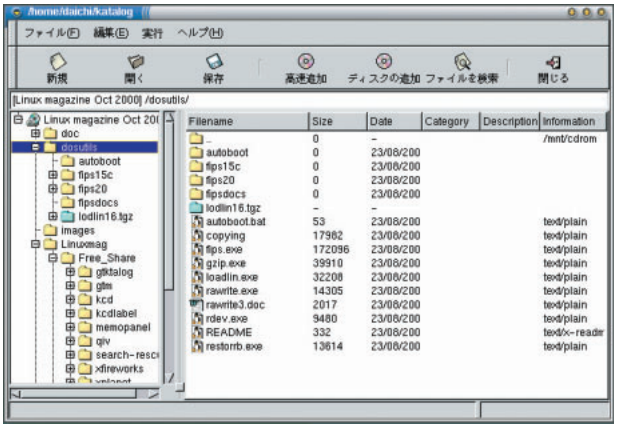
たとえば、ファイル名末尾が「.tar.gz」のtarボールに含まれるファイルは、tarをtzvfオプション付きで起動し、ファイル名は出力の5番目のフィールド…といった設定によって情報を取り出している（**画面3**）。

なおtarコマンドには、bzip2圧縮対応のオプションとして「y」を使うものと、「I」（大文字アイ）を使うものがある。インストールされているtarコマンドがIオプションを利用するものなら、「.tar.bz2」の[プラグインオプション]の設定を、「tIvf」に変更する必要がある。

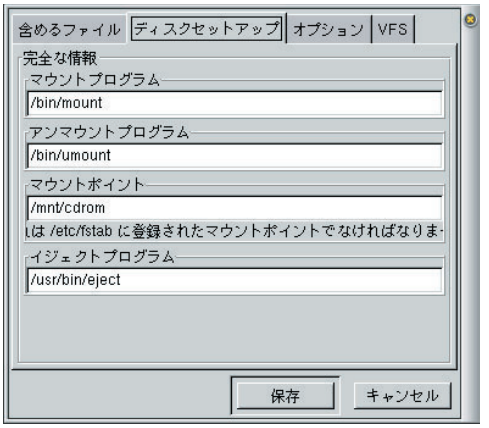
●CD-ROMのファイル情報を登録

GTKtalogを初めて起動した場合は、ツールバーの[新規]ボタンを押してディスクカタログを新規作成しよう。次回からは、ハードディスクに保存したカタログファイルを[開く]ボタンで読み込めばいい。カタログファイルは独自形式で保存される。

ディスクカタログには、複数のCD-ROM（あるいはその他のメディア）のファイル情報（ディレクトリ構造を含む）を登録できる。登録方法としては、次の2つが用意されている。



画面1
ディレクトリツリーとファイル一覧が表示される。



画面2
設定ダイアログでCD-ROMのマウントポイントなどを確認。

（1）CD-ROM専用の方法

複数のCD-ROMを効率よくカタログ化するには、最初のCD-ROMをドライブに挿入した後で、ツールバーの[高速追加]ボタンを押す。続いて、CD-ROMを区別するための名前（ディスクラベル）を入力しよう。

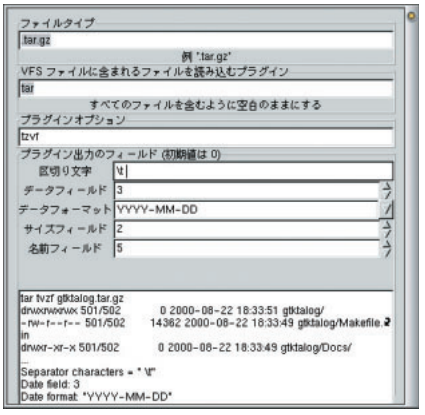
ラベルを入力している間にも、自動的にCD-ROMのマウントが行われ、あらかじめ設定したマウントポイント（初期値は「/mnt/cdrom」）以下のファイル情報がカタログに追加される。進行状況はウィンドウ右下のプログレスバーで確認可能だ。

CD-ROMのすべてのファイル情報を取得すると自動的にイジェクトされるので、次のCD-ROMを挿入して[高速追加]ボタンを押せばいい。以下この繰り返しにより、複数のCD-ROMを効率よくカタログ化できる。

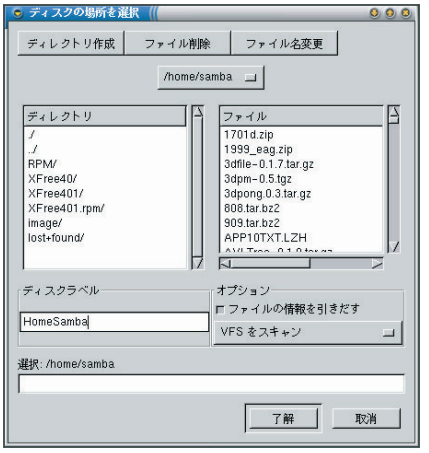
（2）CD-ROM以外にも適用可能な方法

GTKtalogでは、CD-ROM以外のリムーバルディスク（MOやDVD-RAMなど）や、ローカル・リモートのハードディスクのファイル情報をカタログに登録することもできる。

ただし、こちらの方法は、対象となるディスクをあらかじめマウントしておく必要がある。そのため、CD-ROMなら（1）の[高速追加]ボタンを利用したほうが簡単だ。



画面3　VFSで使われるコマンドやオプションを設定する。



画面4　ディスクを判別するラベルと、マウントポイントを指定しよう。

ツールバーの[ディスクの追加]ボタンを押すとダイアログが開くので、ディスクラベルやマウントポイント、VFS利用の有無などを設定する（**画面4**）。[了解]ボタンを押すと、マウントポイント以下のファイル情報がカタログに追加される。

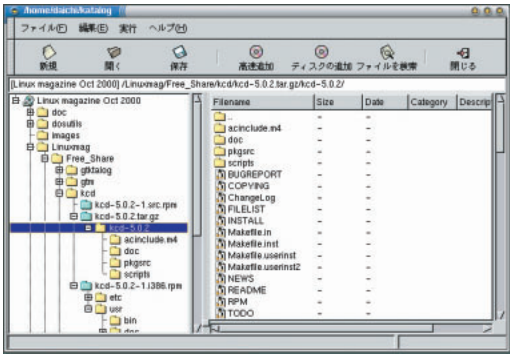
●カタログの閲覧と検索

左側のペインには、ディスクラベルごとのディレクトリツリーが表示される。VFSを有効にして作成した場合は、tarボールやRPMパッケージなどもディレクトリとして表示され、その中に含まれるファイルが右側のペインに表示される（**画面5**）。ディレクトリとVFSはアイコンの色で区別できる。

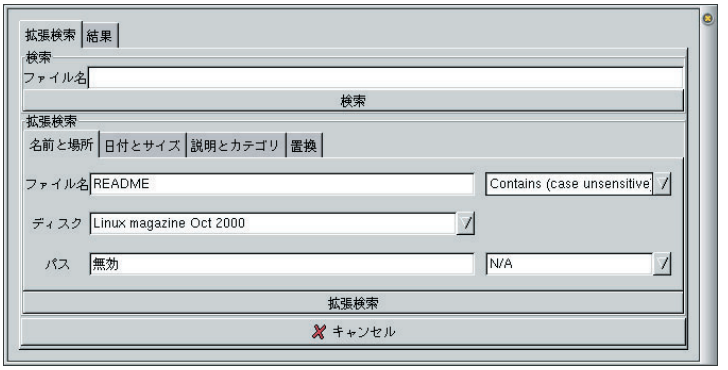
ファイル情報は、右クリックメニューで編集や削除が可能だ。たとえば、編集ではファイル名の変更やカテゴリー／説明の追加を行える。カテゴリーと説明は右側のペインに表示されるほか、検索の際のキーとして利用することも可能だ。

ファイルを検索するには、ツールバーの[ファイルを検索]ボタンを押して検索ダイアログを開く（**画面6**）。ファイル名のみの単純な検索だけでなく、正規表現を利用したファイル名パターンやファイルの日付・サイズ、検索するディスク、VFSも検索対象に含めるかといった複数の条件を組み合わせた拡張検索が可能だ。

検索結果はダイアログの[結果]ページに表示され、ファイル名をダブルクリックすることで、メインウィンドウの表示がそのファイルのあるディレクトリに切り替わる。



画面5　tarボールやRPMパッケージの内容もカタログに登録される。



画面6　複数の条件を組み合わせた拡張検索でファイルを探し出そう。

グラフィカルなSamba用ファイルブラウザ

xSMBrowser

バージョン：3.2　　ライセンス：シェアウェア（30USドル）
家庭内利用はフリー

http://www.public.iastate.edu/~chadspen/homepage.html

xSMBrowserは、Sambaの共有リソースをブラウズするソフトである。smbclientやsmbmountなどのコマンドを内部で実行することで、ネットワーク接続されたWindowsマシンなどのリソースを一覧表示し、マウス操作でファイル転送や閲覧を行える。試用期間60日のシェアウェア（30USドル）だが、家庭内での利用に限っては無料で使用できる。実行にはTcl/TkとexpectkのほかSmaba関連のコマンドが必要だ。

●インストール

xSMBrowserは、tarボールとRPMパッケージの両方が配布されているので、ディストリビューションに合わせて選択しよう。インタプリタのTcl/Tkとexpectkで書かれているのでビルドの必要はない。

tarボールを展開すると、xSMBrowser本体（xsmbrowser）と、アイコン（pixmap/*.gif）が展開される。これらを適当なディレクトリ、たとえば「/usr/local/bin」と「/usr/local/share/pixmaps/xsmbrowser」にそれぞれコピーすればいい。

RPMパッケージのほうは、ファイル名の末尾が「noarch.rpm」になっている。バイナリパッケージと同様にrpmコマンドの-Uvhオプションを使ってインストールしよう。

なお、xSMBrowserは、expectkが/usr/binに置かれていることを前提としている。expectkが別のディレクトリにある場合は、xSMBrowser本体の1行目「#!/usr/bin/expectk -f」を書き換える必要がある。

●起動とリスト表示

「xsmbrowser&」として起動すると、ライセンス表示ウィンドウとともに、アイコン付きのボタンが並んだメインウィンドウが開く（**画面1**）。

tarボールをインストールした場合、最初はアイコンが表示されない。このまま使うこともできるが、[Setup]ボタンで設定ダイアログを開き（**画面2**）、[Image Path]にアイコンのコピー先ディレクトリを設定しよう。再起動すると、ボタンがアイコン付きで表示されるようになる。

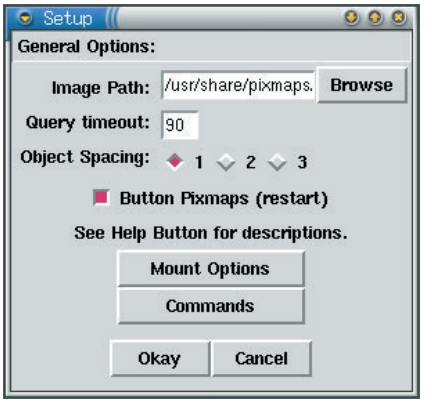
ウィンドウ中央部には、ネットワークのリストが表示されている。初期設定では「Default Samba（broadcast）」と「Default Samba （WINS）」の2つだけだが、ユーザーが任意の設定のネットワークを追加できる。

また、[Workgroups]ボタンを押すとワークグループのリスト、[Favorites]ボタンではユーザーが登録したコンピュータのリストにそれぞれ切り替わる。どのリストも、右クリックメニューで設定変更などが可能だ。

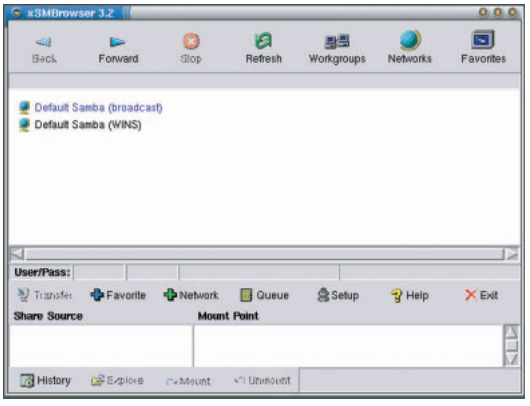
●共有リソースにアクセスする

ネットワークリストの「Default Samba （broadcast）」をクリックするか、ワークグループリストでワークグループ名をクリックすると、コンピュータ名のリストに切り替わる。このように、xSMBrowserでは、ダブルクリックではなく、クリックによる選択操作を採用している。

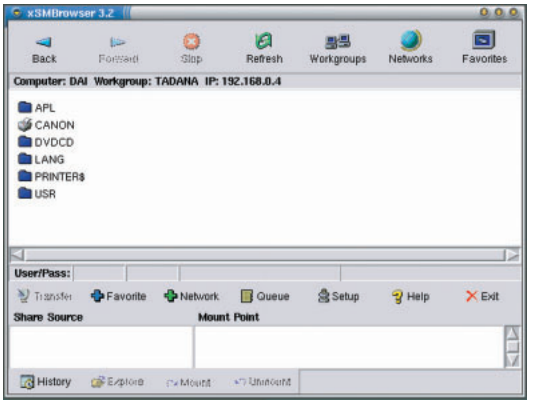
コンピュータ名をクリックすると、今度はそのコンピュータの共有リソースの一覧に切り替わる（**画面3**）。プリンタなどの共有名も表示されるが、



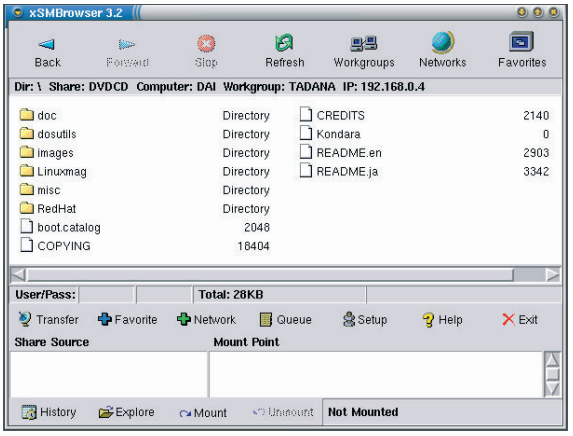
画面2　tarボールからインストールした場合は[Image Path]を設定。



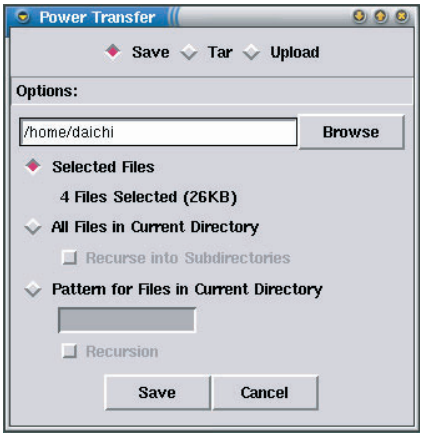
画面1　アイコンを利用した各種ボタンが並ぶメインウィンドウ。



画面3　選択したコンピュータの共有リソースが一覧表示される。



画面4　smbclientが取得した情報に基づくファイル一覧表示。



画面5　ファイル転送やアーカイブはこのダイアログで行う。

xSMBrowserから利用できるのはディレクトリだけだ。

ディレクトリの内容を表示するにはいくつかの方法がある。通常は、ディレクトリの共有名をそのままクリックすればいい。smbclientが自動的に実行され、取得したファイル情報に基づいてファイルやディレクトリが一覧表示される（**画面4**）。

ファイル一覧画面では、[Transfer]ボタンで開くPower Transferダイアログを使って、選択したファイル（あるいはディレクトリ内の全ファイル）のセーブやアーカイブなどを行える（**画面5**）。また、右クリックメニューの[Open file]では、内蔵ビューアによるテキスト表示（**画面6**）や、Gimpなどによる閲覧が可能だ。

●共有リソースをマウントする

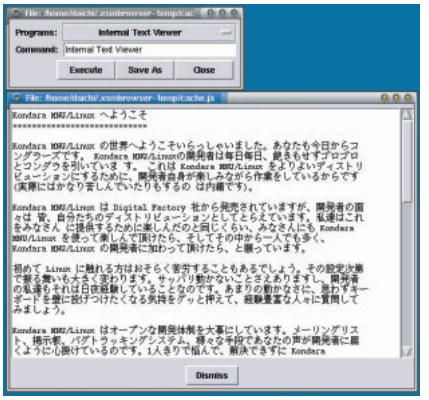
smbclientの代わりにsmbmount／smbumountを利用して、指定したマウントポイントに共有リソースをマウントすることもできる。

ただし、一般ユーザーの場合は、smbmount／smbumountにsuidビットを立てる必要がある（rootになってchmodコマンドの+sオプションで設定）。suidビット付きのコマンドはセキュリティの弱点になりやすいので、各自の環境を考慮した上で実行されたい。

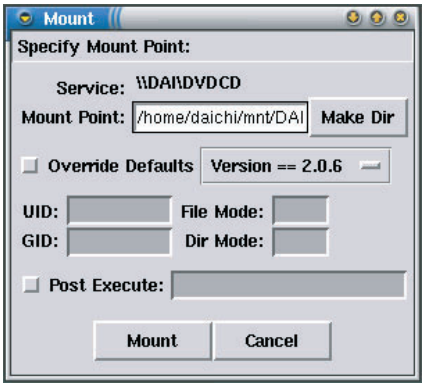
共有名を右クリックして、[Mount Share]を選択するとダイアログが開く（**画面7**）。マウントポイント（初期値は「~/mnt/コンピュータ名/共有名」）などを設定した後で、[Mount]ボタンを押してマウントしよう。

成功すると、ウィンドウ下部領域にマウント情報が表示される。[Explorer]ボタンでMidnight Commanderを起動してマウント先をブラウズしたり（**画面8**）、[Unmount]ボタンでマウントを解除することも可能だ。

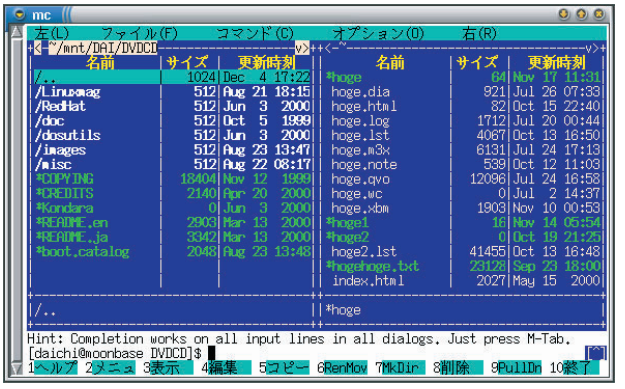
なお、国産ディストリビューションに付属するMidnight Commanderは日本語化されているため、xtermでは文字化けしてしまう。設定ダイアログの[Mount Options]ボタンを押してマウント設定ダイアログを開き、[Post Execute]の最初の「xterm」を「rxvt」に変更するといいだろう。



画面6　内蔵ビューアで共有先のテキストファイルを閲覧。日本語もOKだ。



画面7　smbmountを利用して共有リソースをマウントすることも可能。



画面8　rxvt上のMidnight Commanderでマウントポイントをブラウズ。

パーティションの作成やリサイズを行う

GNU Parted／nParted

バージョン：1.4.4　　ライセンス：GPL

http://www.gnu.org/software/parted/
http://quark.fe.up.pt/laespiral/software/（nParted）

GNU Partedは、パーティションやファイルシステムの作成／リサイズ／コピー／移動などを行うコンソールベースのソフトである。ファイルシステムはext2、linux-swap、FAT16／32に対応している。実行にはカーネル2.2以降とglibc 2.1以降が必要だ。また、カーソルキーなどを使ってより簡単に操作を行えるフロントエンド「nParted」もあわせて紹介する。

●ビルドとインストール

GNU Partedは、ソース一式をtar＋gzipしたtarボールのみ配布されている。ビルドとインストールは、「./configure」→「make」→「make install」という一般的な手順だ。

ビルド中に「libuuidが見つからない」といったエラーが表示される場合は、e2fsprogsパッケージが古いことが考えられるので、バージョンアップしてから再度ビルドを行おう。

また、tarボールに含まれるSPECファイルを利用して、RPMのソース／バイナリパッケージを作成することも可能だ。「rpm -ta parted-1.4.4.tar.gz」とすればいい。

続いて、nPartedのtarボールを展開して、ビルドとインストールを行う。手順はGNU Partedと同じだ（ビルドにはnewtが必要）。SPECファイルが付属しないため、RPMパッケージ化は面倒だ。そのまま/usr/local以下にインストールしよう。

●GNU Partedを使用する前に

GNU Partedは、誤った操作を行った場合でも、データ損失をできるだけ最小限にとどめるように設計されている。しかし、ユーザーの操作によっては必要なファイルシステムを誤って削除してしまい、重要なデータを永久に失ってしまう危険もある。以下の作業はユーザー各自の責任で行ってほしい。できれば、CD-Rなどを使ってOSをまるごとバックアップしておき、万が一の際にも復活できるようにしておくとよいだろう。

GNU Partedには日本語ドキュメントが用意されている（本誌付録CD-ROMにも収録）。まずはこれを読むことを強くお勧めする。ただし、少し前のGNU Partedに基づいた記述になっているので、各コマンドの詳細などについてはtarボールに含まれる英語ドキュメント（USER）を参照されたい。

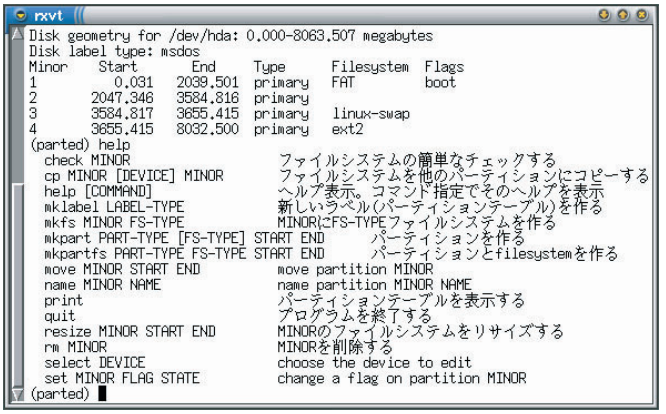
もし、パーティションやファイルシステムに関する基本的な知識が不足しているようなら、「Linux Partition HOWTO」（JFプロジェクトによる日本語訳は、http://www.linux.or.jp/JF/JFdocs/Partition.html）を先に読むとよいだろう（特に第2章）。

●GNU Partedをそのまま使う

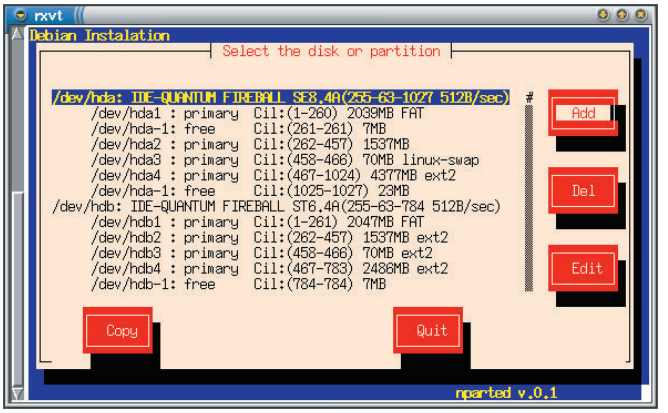
すでに述べたように、パーティションの操作には危険が伴うため、GNU Partedの実行にはスーパーユーザー（root）権限が必要だ。コマンドラインで「su -」として、スーパーユーザーになって作業しよう。

コマンドラインで「parted」とすると、GNU Partedが起動して、メッセージとプロンプト（parted）が表示される。付属の日本語カタログにより、日本語環境ではヘルプや一部のメッセージが日本語表示される（**画面1**）。英語表示にするには「LANG=C parted」として起動すればいい。

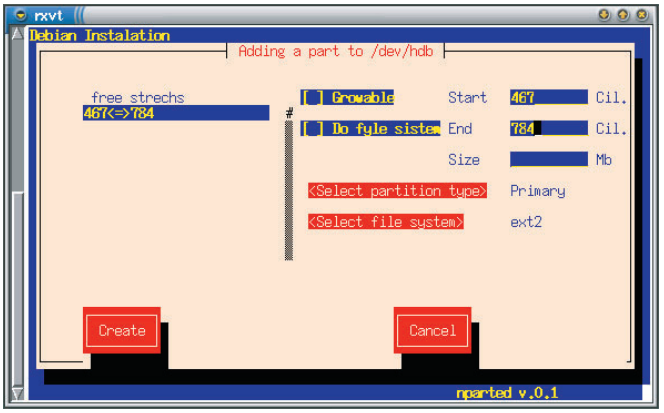
操作はコマンド入力方式で、「print」などのコマンド（**表1**）を入力することで、表示／作成／削除／リサイズなどの処理を行う。今回はnPartedによ



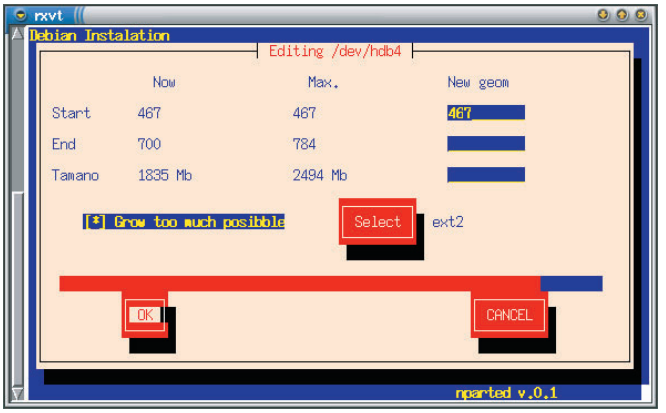
画面1　GNU Partedはコマンド入力方式のコンソール系ツールだ。



画面2　nPartedでは、カーソルキーなどを使った操作が可能。



画面3　空き領域に新しいパーティションを作成する。



画面4　既存のパーティションのサイズを変更することも可能だ。

る操作を中心に説明するので、これらのコマンドの詳細はドキュメントを参照してほしい。

なお、GNU Partedでは、現在マウントしているパーティションに対してはリサイズなどの操作を行えないので注意しよう。特に、実行中のLinuxのルートパーティションはアンマウントもできないため絶対に変更できない。こうした場合は、あとで述べるGNU Partedブートディスクを利用して操作する必要がある。

●nPartedを利用する

フロントエンドのnPartedを利用すると、もう少しグラフィカルな操作が可能だ。スーパーユーザーになった状態で「nParted」とすると、コンソール（あるいは端末画面）にnewtライブラリを利用したリストやボタンが表示される（**画面2**）。

リストには接続されている全ドライブのパーティション情報がまとめて表示され、5つのボタンでパーティションの追加や削除、編集（リサイズ）、コピー、終了などの操作を行う。項目間の移動はTabキー、リストのスクロールは↑↓キー、ボタンを押すにはスペースキーを利用する。

たとえば、パーティションとファイルシステムの作成を行うには、リスト中の「free」と表示された空き領域にカーソルを合わせておいて、[Add]ボタンを押す。すると、追加用の画面に切り替わるので（**画面3**）、開始・終了シリンダ、パーティションとファイルシステムの種類などを設定して[Create]ボタンを押せばいい。

同様に、既存のパーティションにカーソルを合わせて[Edit]ボタンを押すと、リサイズを行える（**画面4**）。現在のところ、ext2ファイルシステムでは開始シリンダは変更できず、終了シリンダのみ変更可能なので注意されたい。「Grow too much posibble」（綴りが違うがこのように表示される）をチェックすれば、可能な限りの容量を確保してくれる。

●GNU Partedブートディスクを使う

動作中のLinuxマシンのルートパーティションなどは、GNU Partedで直接操作することはできない。このような場合には、GNU Partedブートディスクを作成し、マシンをフロッピーディスクからブートさせることで実行可能になる。

空きフロッピーディスクを1枚用意し、ddコマンドを利用してブートイメージ（partboot.img）を書き込む。具体的には「dd if=partboot.img of=/dev/fd0 bs=1440k」とすればいい。

このブートディスクで起動すると、最小構成のLinuxが起動して、bashのプロンプトが表示される。

あとは、GNU Partedを起動して、パーティションなどの操作を行えばいい。このほか、mountやls、moreといった最低限のコマンドが用意されている。ただし、カーネルにUDMA66対応パッチが当たっていないため、使用しているドライブやマザーボードのIDEインターフェイスによっては、認識されない可能性もある。

なお、ルートパーティションなどLinuxのカーネルが含まれるパーティションのリサイズや移動などを行った場合は、/sbin/liloを実行する必要がある。具体的な操作はブートフロッピーのREADMEを参照されたい。

コマンド	動作
check	ファイルシステムの簡易検査
cp	ファイルシステムのコピー
help	一般的な（あるいは指定したコマンドの）ヘルプを表示
mklabel	新しいパーティションテーブルを作成
mkfs	ファイルシステムを作成
mkpart	パーティションを作成
mkpartfs	ファイルシステムと一緒にパーティションを作成
move	パーティションを移動
name	パーティションに名前を付ける
print	パーティションテーブルを表示
quit	GNU Partedを終了
resize	ファイルシステムのリサイズ
rm	パーティションを削除
set	パーティションのフラグを変更

表1　GNU Partedのコマンド一覧

OpenGLを使った美しいシューティングゲーム

Chromium B.S.U.

バージョン：0.9.8a　　ライセンス：Artistic

http://www.reptilelabour.com/software/chromium/

Chromium B.S.Uは、OpenGLを利用して3Dレンダリングされた、美しい縦スクロールシューティングゲームだ。アイテムによるパワーアップやボスキャラはお約束だが、しだいに消耗する武器やシールドの扱い、敵キャラを逃してはいけない点、自爆によるボム効果など、国産シューティングとのテイストの違いも興味深い。実行には3Dライブラリ MesaとSDL 1.1.8以降、3Dアクセラレータ環境が必須だ。

●ビルドとインストール

Choromium B.S.U.は、データ／ソース／バイナリのtarボールがそれぞれ配布されている。実行にはバイナリ／ソースのいずれか一方とデータが必要だ。ソースの場合は「make」とするだけで、展開先のbinディレクトリに実行ファイル（chromium）が作成される。一方、バイナリの場合は展開先のbinディレクトリに実行ファイルが展開される。これを/usr/local/binなどに手動でコピーしよう。

データのほうは、適当なディレクトリで展開し、環境変数CHROMIUM_DATAにデータディレクトリを設定する。たとえば、/usr/local/shareに展開した場合は、「export CHROMIUM_DATA=/usr/local/share/Chromium-0.9/data」とすればいい。

●マウスで自機を操作する

起動すると、3Dレンダリングされたタイトル画面が表示される（**画面1**）。ここでは、ゲームの難易度や画面効果、画面サイズなどをカーソルキーで設定可能だ。初めは、難易度を「whimp」に設定するといいだろう。

プレイ中はマウスを使って自機を移動させ、左ボタンで武器を発射する（**画面2**）。武器は3種類あり、3色の骸骨のアイテムを拾うと使えるようになる。左ボタンを押し続ければ連射されるが、使った分だけ消耗してしまうので（残量は左上のグラフに表示）、むやみな連射は控えたい。

なお、敵キャラが画面下に到達すると、それだけで自機が一機失われてしまう。シールド（左右下側に表示）が残っていれば、敵や弾に触れても自機は破壊されないので、体当たり戦法も有効だ。シールドは3色のペンギンのアイコンを拾うと補充される。

もし、複数の敵に防衛線を突破されてしまいそうなら、右ボタンのダブルクリックで自爆し、爆風で敵を一掃するという最後の手段を使おう（自機は1機減ってしまうが）。

各面の最後に登場するボスキャラ（**画面3**）を倒せば、シールドが補充されて次の面に移る。敵キャラの種類はそれほど豊富ではないが、十分に熱い戦いを楽しめるはずだ。



画面1　画面表示とBGMがクールなタイトル画面。難易度設定も可能だ。



画面2　画面上から登場する敵をすべて破壊せよ。体当たりも有効だ。



画面3　各面の最後には、巨大なボスキャラが襲いかかってくる。

玉を指定された配置に揃えるパズルゲーム

Marbles

バージョン：001126　　ライセンス：GPL

http://lgames.sourceforge.net/

Marblesは、制限時間内に複数の玉を決められた配置に揃えるパズルゲームだ。玉は障害物にぶつかるまで指示された方向に動き続けるため、それぞれの玉を動かす順番を考えないと、すぐに手詰まりになってしまう。ルールは単純だが思考力の要求されるゲームだ。時間切れが近づくとサウンドエフェクトが緊迫感を盛り立てる。実行にはゲームライブラリのSDL 1.0.8以降が必要だ。

●ビルドから実行まで

Marblesは、ソース一式をtar＋gzipしたtarボールのみ配布されている。ビルドとインストールは「./configure」→「make」→「make install」という一般的な手順だ。

「marbles&」として起動すると、タイトル画面が表示される。マウスで[New Game]をクリックしてスタート画面に切り替えよう（**画面1**）。

初めてプレイする場合は、スタート画面から[Edit Profiles]－[Create Profile]と進んで、あなたの名前を設定する必要がある。スタート画面に戻ったら、その名前が表示されるまで[Profile]をクリックし、[Start]をクリックしてプレイを始めよう。



画面1
あなたのプロファイルを登録してからゲームを始めよう。

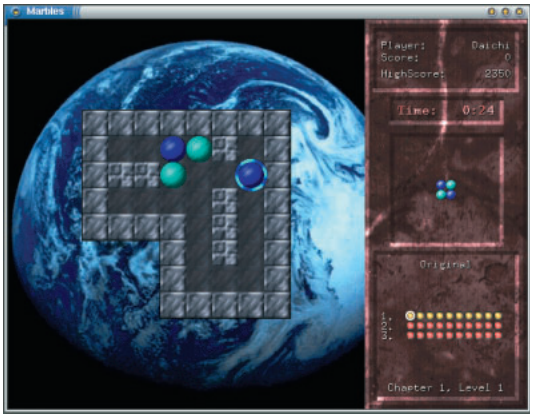
●制限時間内に玉を配置せよ

プレイ中のウィンドウには、壁に囲まれたフィールドと、何種類かの色と模様を持つ玉が表示される（**画面2**）。ルールは単純で、右のパネルに表示されているのと同じ配置になるように、玉を動かして揃えればいい。

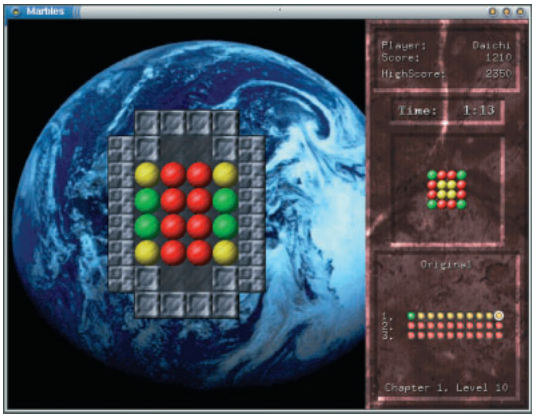
操作にはマウスを使用する。まず、動かしたい玉をクリックで選択すると、その玉の周囲に円形のカーソルが表示される。続いて、玉の四方いずれかをクリックすると、そちらに玉が移動する。玉は、壁か他の玉と衝突するまで止まらないので、完成時の玉の配置を考慮して、どの玉から動かしたらいいか計算しよう。

残り時間がなくなるまでにすべての玉を揃えられなければゲームオーバーで、そのレベル（面）は最初からやりなおしだ。一方、時間内にすべての玉を揃えると玉が粉々に砕けるグラフィックが表示され、スコアの計算に切り替わる。残り時間はボーナスとして加算されるので、できるだけ素早く揃えたほうが得だ。クリックすると次のレベルのプレイが始まる。

先に進むにつれ、玉の種類が増えて配置が複雑になり（**画面3**）、テレポーターなどのギミックも登場する。3つのチャプターごとに10個のレベルが用意されており、チャプター内ではレベルを飛ばしてプレイすることも可能だ。10レベルすべてをクリアすると次のチャプターに進める。



画面2
右側に示された配置になるようにマウスクリックで玉を動かせ。



画面3
3つのチャプターそれぞれに10レベル（面）用意されている。

文字列の送出も可能なソフトキーボード

GTKeyboard

バージョン：1.1.5　　ライセンス：GPL

http://opop.nols.com/gtkeyboard.html

GTKeybaordは、マウスでキー操作を行うソフトキーボードだ。任意のアプリにキーコードを送れるほか、編集可能なテキストバッファも内蔵している。キーボードレスのマシンや、複数のキーの同時打鍵が困難なユーザーでも複雑なキー操作が可能だ。HTMLのタグやユーザー定義文字列をまとめて送出できるので、キーボードが接続されているマシンでも利用価値がある。動作にはGTK＋が必要だ。

●ビルドから起動まで

GTKeyboardの最新版は、ソース一式をtar＋gzipしたtarボールのみ配布されている。ビルドとインストールは、「./configure」→「make」→「make install」という一般的な手順だ。

また、tarボールに含まれるSPECファイル（rpm/gtkeyboard.spec）を利用してRPMパッケージを作成することも可能だ。ただし、いったんtarボールを展開して、SPECファイルの5、9行目の「0.99」を「1.1.5」に変更する必要がある。修正したSPECファイルは/usr/src/redhat/SPECS、tarボールは/usr/src/redhat/SOURCESにそれぞれコピーし、「rpm -ba /usr/src/redhat/SPECS/gtkeyboard.spec」とすると、RPMソース／バイナリパッケージを作成できる。

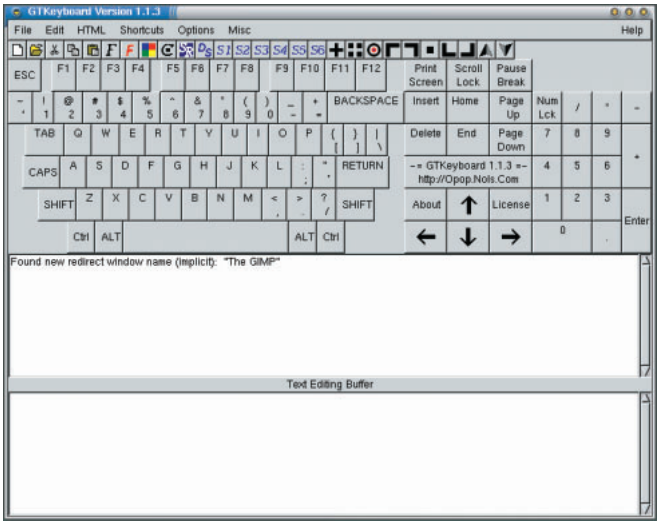
●柔軟なカスタマイズが可能

「LANG=C gtkeyboard&」として起動すると、101英語配列のキーボードが描かれたウィンドウが開く（**画面1**）。キーボードの上にはツールバー、下にはステータス表示エリアと内蔵テキストバッファが並んでいる。

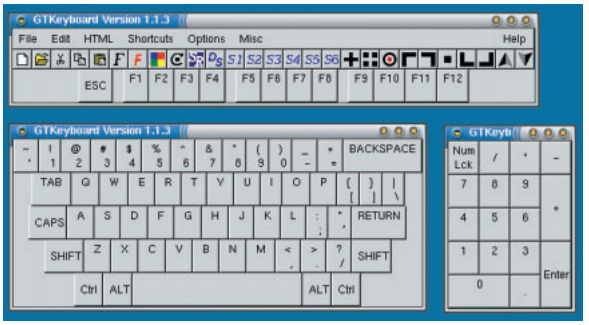
各部の表示は[Options]－[Looks]以下のメニューでカスタマイズできる。フルキーやテンキー、ツールバーなどだけ表示することも可能だ（**画面2**）。また、[Options]－[Choose a Layout File]で、キーボードレイアウトファイル（拡張子.key）を選択すれば、キー配列そのものを変更できる。

使い方は簡単で、キーコードを送出したいアプリのウィンドウをクリックしてから、GTKeyboardのキーをマウスで押せばいい。CtrlキーやAltキーは一度押すとロックする。アプリの中には、ktermやXEmacsなどアプリ側での設定が必要なものもある。詳細は付属のNEWSを参照されたい。

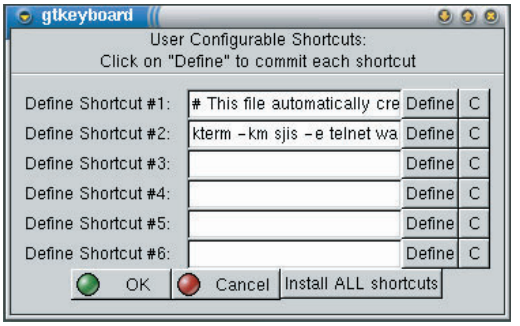
このほか、[HTML]メニューにはHTMLで利用する各種タグ、[Shortcuts]メニューにはさまざまなショートカットが用意されており、メニューを選択するだけで文字列が送出される。ユーザーが内容を自由に定義できるショートカットは6つ用意されており（**画面3**）、メニューのほかツールバーのボタンからも送出可能だ。



画面1
初期設定ではUS配列のキーボードとテキストバッファなどを表示。



画面2
キーボード各部の表示は自由にカスタマイズできる。



画面3
ユーザーが自由に内容を定義できるショートカットを用意。

自動スクロールするテキストビューア

Reed

バージョン：4.5　　ライセンス：GPL

http://www.sacredchao.net/software/reed/

Reedは、ktermなどの端末画面でテキストの内容を閲覧するテキストビューアだ。日本語の表示も可能だ。最大の特長は、いちいちキーを押さないでもテキストの内容を読み進められる自動スクロール機能。スクロール速度は何段階かに変更でき、一時停止や再開も可能。もちろん、キー操作によるスクロールや検索、ブックマークなど、通常のテキストビューアとしての機能も用意されている。

●ビルドと実行

Reedは、ソース一式をtar＋gzipしたtarボールのみ配布されている。ビルドとインストールは、「./configure」→「make」→「make install」という一般的な手順だ。

Reedを起動するには、ktermなどのコマンドラインで、「Reed テキストファイル名」とすればいい。複数のファイル名を並べたり、ワイルドカードを利用することも可能だ。

●自動スクロールでらくらく閲覧

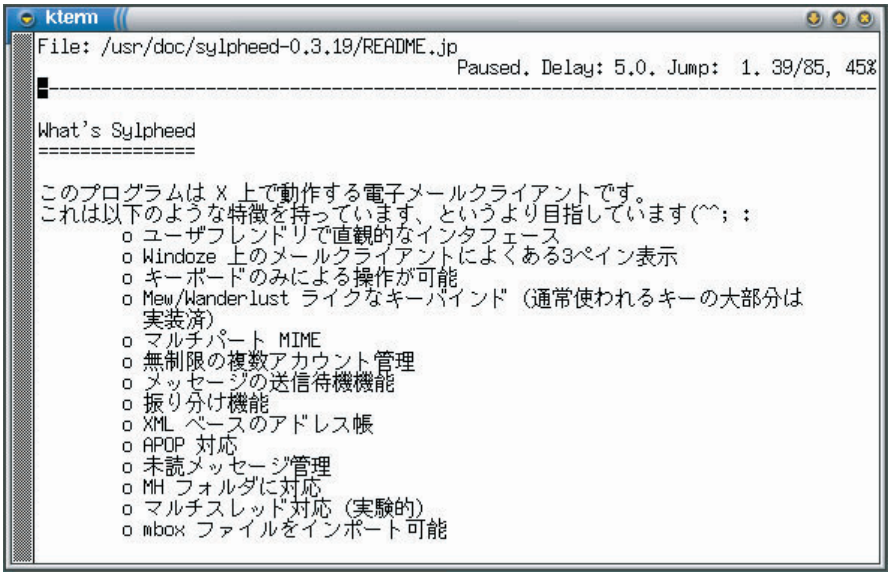
画面の上部にはファイル名や行数などの情報が並んでおり、1行の水平線で区切られてテキストファイルの内容が表示される（**画面1**）。

初期設定では、Reedを起動すると、テキストは自動的にスクロールする（pキーで停止／再開）。1行のスクロールに要する遅延時間（1/10秒単位）の初期値は5なので、0.5秒ごとにスクロールしていくことになる。この設定はReed実行中にも次の2通りの方法で変更できる。sキーを押すと、数字で遅延時間を直接指定できる。またフルキー上の＋／－キーを押せば、約1/30秒単位で、遅延時間を増減させることができる。自然に読める速度に調整すれば、テキストの内容を一切キーに触れずに閲覧することも可能だ。

複数のファイルを開いている場合は、：キーに続けてp／nキーを押すと前後のファイルに切り替わる。Reed実行中に新たなファイルを読み込む（あるいは実行する）ことも可能だ（：キーに続けてeキー）。

このほか、less風のキー操作によって、スクロールや正規表現検索を行えるなど、テキストビューアとしての機能も充実している（**表1**）。

特に、後で戻りたい箇所にマークを付けるブックマーク機能に関しては、less風の1文字マーク（「"」または「'」キー）だけでなく、25文字までの意味のある名前を付けられる文字列ブックマーク（M／mキーを利用）が用意されているため、lessよりも使いやすくなっている。



画面1　自動スクロールにより長いテキストをゆっくり閲覧できる。

コマンド	動作
q	Reedを終了して画面クリア
p	スクロールを一時停止・再開
s	遅延時間（1/10秒単位）を設定
+、-	遅延時間を増加/減少
j	スクロール行数を設定
t	指定した行に移動
%	指定した割合（%単位）に移動
/、?	前方/後方に正規表現検索
↓、e	1行スクロール
↑、y、k	1行逆スクロール
d	半ページスクロール
u	半ページ逆スクロール
PgDn、Space、z	1ページスクロール
PgUp、w、b	1ページ逆スクロール
Home、<、,	ファイルの先頭に移動
End、>、.	ファイルの末尾に移動
Enter	遅延時間無視のスクロール
M	文字列ブックマークを設定
m	文字列ブックマークに移動
"	1文字ブックマークを設定
'	1文字ブックマークに移動
D	指定したブックマークを削除
C	ブックマークを全て削除
!	シェルコマンドの実行
r	画面を再描画
v	エディタで編集
:n	次のファイルに切り替え
:p	前のファイルに切り替え
:d	ファイルリストから削除
:e	ファイルリストに追加
:r	再度読み込み（桁数調整）
:q	Reedを終了

表1　Reedのキー操作

Software Catch Up

国産DOSワープロの歴史を受け継ぐ

dp/NOTE for Linux/BSD Ver2.0

dp/NOTEは、UNIX上のワープロとして長い歴史を持っている純国産の日本語ワープロソフトだ。Linuxだけでなく、多くのPC UNIXで動作するという、幅広いプラットフォームが魅力でもある。

文：塩田紳二
Text : Shinji Shioda

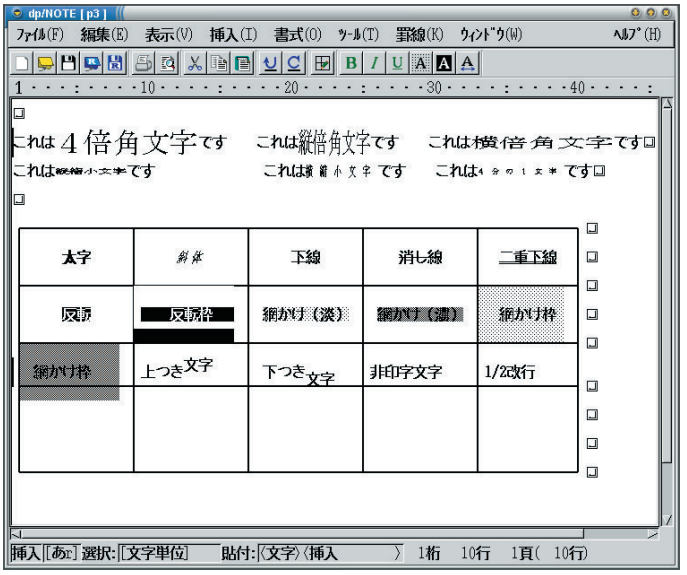
価格	9800円
問い合わせ先	オムロンソフトウェア 044-246-6006 http://www.omronsoft.co.jp/

dp/NOTE（**画面1**）は、オムロンソフトウェアが発売するワープロソフトである。PC UNIX用としてLinuxだけでなく、FreeBSDやNetBSD、BSD/OSなどにも対応する。

dp/NOTEは、かつてワークステーションが急成長した時代（1990年代前半）に、ワープロソフトとして登場したもので、当時はオムロンの販売するワークステーション（LUNA）などで動いていた。また、現在でも図形を扱うことができるdp/NOTE-図形というパッケージが、SolarisやIRIX、HP-UX用など向けに販売されている。

オムロンソフトウェアは、Linux系ディストリビューションには必ずといっていいほど収録されている、Wnnの開発元でもある（開発当時はオムロン）。同社が開発したWnnは、パッケージ製品としてWnn6を販売しているほか、かつてのWnn4をベースにしたFreeWnnなどが広く使われている。そういうわけで同社は、国内UNIX業界では歴史のある著名なメーカーでもあるといえる。

さて、このdp/NOTEはPC UNIX用ということもあり、個人向けに9800円（日本語TureTypeフォントを含む）と低価格で販売されており、Linux用のパッケージソフトながら入手は比較的容易である。



画面1　dp/NOTEはLinuxやFreeBSDに対応した国産ワープロソフトだ。

パッケージは、CD-ROMとインストール方法などを記載したマニュアルから構成されている。また、Microsoft Word用とはRTF形式を介して相互に文書の交換が可能となっている。

なお、dp/NOTEのメニューから実行する各種の変換機能などは、単独のフィルタプログラムとなっている。

さらにdp/NOTEには、POPを使ったメールクライアントの機能が含まれており、付属アプリケーションとしてメールの到着を知らせるdpbiffが同梱されている。

インストール

dp/NOTEのインストールは、最近のソフトウェアにしては少々難しい部類に属する。Red Hat系ではRPMを使ったインストールが可能なものの、ライセンスサーバや日本語入力関連の設定をインストール後に手作業で行う必要があるからである。

余談だが、付属のマニュアルを見ていたら、かつて筆者がワークステーシ

ョン関係の仕事をしていた頃を思い出してしまった。当時は、パッケージアプリケーションといえども、いまのWindowsアプリケーションのような自動インストールはできず、シェルスクリプトで半分は自動でインストールして、あとは手作業でprintcapやフォント、Xのリソース設定などを行ったものである。これは、メーカーごとに設定ファイルなどに違いがあり、一番確実なのは環境を熟知している管理者が自分でやることだったからである。ひどいものになると、OSのバージョンが指定されていて、勝手に/etc/rcファイルを置き換えるものさえあった。また、今のようにデスクトップ環境も充実しておらず、ウィンドウマネージャは、twmやMotifなどで、メニューへの登録などはかなり難しい問題だった（アプリケーション側から、X Window Systemの環境を特定する一般的方法がないため）。このため、多くの場合、起動はktermなどからコマンドラインで指定したものであった。

dp/NOTEではLinux系のパッケージとして、Slackware Package形式、Red Hat RPM形式、Debian deb形式がサポートされている。筆者の環境では、RPMを使ったインストールを行った。RPMでのインストールでは、インストール作業のかなりの部分が自動化されるが、それでもrpmコマンドでパッケージをインストールしたのち、以下の作業を行う必要がある。

- **ライセンスサーバ関連の設定**
- **ライセンス情報の設定**
- **日本語入力メソッドの設定**

dp/NOTEは、ライセンスサーバ（dpkeyserv）を使って動作するため、たとえ1人で使う場合でもライセンスサーバを動かす必要がある。なお、このライセンスサーバは、Wnn6など、オムロンソフトウェアから発売されているほかのソフトウェアでも利用しているので、すでにそれらを導入していれば設定する必要はない。

RPMでインストールした場合、このdpkeyservが指定ディレクトリにコピーされるだけで、/etc/rc.d/init.dなどにサーバ起動用のエントリが作成されない。このため、ユーザーがdpkeyservをシステム起動時に起動するための作業を行う必要がある。マニュアルでは、rc.localに4行ほどのスクリプトを手動で追加するように指示されているが、ここは、あらかじめメーカー側で起動用のスクリプトを用意して、chkconfigなどで設定してほしいところである。

もう1つは、CD-ROMに格納されているライセンス情報を、ハードディスクにインストールされたライセンス情報ファイルに追加するという作業が残っている。これは、dpkeylistというファイルに正式ライセンス用の情報をコピーするというものである。これもやはり手作業でコピーすることとなっているが、このあたりもライセンスサーバが管理用コマンドを持ち、自動で行ってほしいところだ。ユーザーにライセンスによる管理を強制するなら、それで手間をかけさせるというのは少々問題を感じる。

最後に日本語入力メソッドの設定を行う。これはdp/NOTEの標準日本語入力メソッドが、デフォルトでWnn6（パッケージ版のWnn）になっており、jserverと直接交信するためだ。

現状ではkinput2経由でcannaなり、FreeWnnなどを使う方式が各ディストリビューションで標準的に採用されているため、多くの場合この切り替えを

Column

FreeWnnをRed Hat系で使っている場合

現在、多くのディストリビューションには、FreeWnnが組み込まれているが、ほとんどの場合、kinput2を併用した形式で、jserverを直接使うとか、uumを使う設定にはなっていないことが多い。

しかしdp/NOTEでは、商用版であるWnn6を使用することを前提としているため、標準状態では直接jserverとの接続を行うようになる。このため、これ以外を使うにはちょっとした設定が必要となる。というのは、Wnn関係の各種ファイルをdp/NOTEが直接参照しているからである。

起動時に必要だとメッセージが出るのは、

```
uumrc
jserverrc
```

などのファイルである（ただし、実際にはuumrcなどでさらに別のファイルを参照しているため、ほかのファイルも必要）。これらのファイルは、Red Hat系では/etc/FreeWnn/jaに置かれているが、Wnn本来の位置は/usr/local/lib/wnn/ja_JPであり、dp/NOTEはこちらの位置にあることを期待している。このため、そのままではjserverが起動していても接続が行えない。

最も簡単な方法は、/usr/local/lib/wnn/ja_JPを/etc/FreeWnn/jaのシンボリックリンクとして作成することである。こうすることで、dp/NOTEは各種ファイルをちゃんと読み込めるようになる。

特にuumrcでは、ほかのファイルを指定しており、ユーザーのホームディレクトリに.uumrcとしてコピーした場合には、ディレクトリパスなどを書き換える必要が出てくる。

最近では、uumなどを使うことは少ないと思われるので、uumrcなどをホームディレクトリに置いておいても、特にメリットはないだろう。

行う必要がある。

これには、各ユーザーのホームディレクトリに「.dprc」というファイルを作り、そこに、

```
use_fep_name  xim_root
```

または、

```
use_fep_name  xim_over
```

という記述を入れる必要がある。またjserverと直接交信させる場合でも、jserverなどの設定はWnn6に準拠したものになっているため、FreeWnnなどをRed Hat系で使っている場合、ちょっとした設定が必要となる（前ページ**コラム**参照）。

手順として最も簡単なのは、「.dprc」を設定する方法であった。jserverを使う設定は、筆者がFreeWnn以前のWnnを使ったことがなかったので、本来のファイル位置など知らず、調べるのが大変だった。というのは、Red Hat系では、jserverrcなどを/etc/FreeWnn以下に置いているからである。幸い、FreeWnnに含まれているmanページが元のままだったため、これを見ることでdp/NOTEが想定しているフォルダを特定することができた（このあたりについてもコラム参照）。

もっとも、古くからのユーザーであれば、こうしたことはなんでもないことだとは思うが、最近Linuxを始めたばかりというユーザーなどにはちょっと難しいという気がする。比較としていえば、多くのLinux用オフィス系アプリケーションがインストーラを起動する程度の簡単なインストールを実現しているなか、こうした手作業の多いインストールは、かなりデメリットに見える。

GUI

dp/NOTE 2.0は、GTK+を採用してGUIを構築している。機能としてはワープロのみなので、ウィンドウは1つしかない。ただし、1つのウィンドウを上下に分割して別々の文書を編集することができる（**画面2**）。このほか、複数のdp/NOTEを起動して別々のファイルを編集することが可能である。なお、dp/NOTEのライセンスは、Xの1つのディスプレイ内であれば、いくつ起動しても、複数ユーザーが起動しても1ライセンスとして扱われる。

ウィンドウは、上部にメニューバー、ツールバーを持ち、下部にステータスバーを持つ。また、ツールバー直下にはルーラが配置されている（**画面3**）。

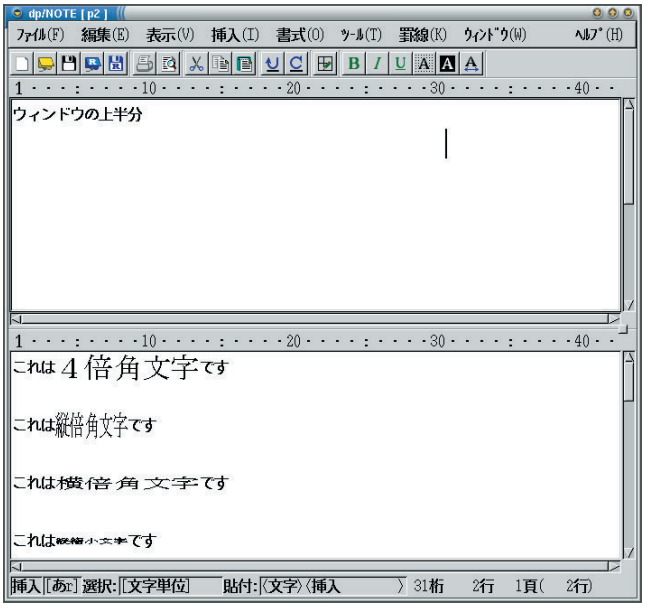
文章を編集するエリアでは、右ボタンで範囲選択の指定ができる。これは、マウスをドラッグして指定した選択範囲を、文字単位、行単位、矩形ブロックにするかを選ぶものだ（**画面4**）。なお、マウスホイールには対応していないようである。

ステータスラインには、jserverと接続しているときには、かな漢字変換の入力モードが表示され、現在の範囲選択モードや、状態、カーソル位置などが表示される。

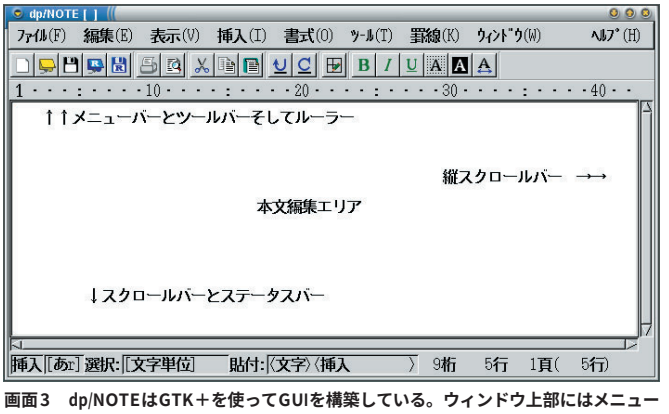
左右のマージンや文字数、行数などについては、印刷メニュー内で設定し、それが画面に反映されるようになる。ルーラはこのときも文字数を表すものだが、ここでGUI操作によりマージンを指定するなどの機能はない。

電子メールは専用のウィンドウから利用する。これはメッセージリストと本文表示エリアからなるもので、こちらもツールバーとメニューバーを備えている。

dp/NOTEで利用する各種の機能キーは、カスタマイズが可能である。このため、カーソル移動や文字削除など



画面2　ウィンドウを上下に分割して2つの文書ファイルを同時に編集することができる。



画面3　dp/NOTEはGTK＋を使ってGUIを構築している。ウィンドウ上部にはメニューバーやツールバー、ルーラーがあり、下部にはステータスバーがある。

を、使い慣れたエディタと似たものにすることができる。このほか、ウィンドウタイトル、編集領域の白黒反転や文字色、背景色指定、スクロールバーの有無、表示用フォントの指定などの表示関連のカスタマイズも可能になっている。ただし、これらは外部の初期設定ファイル（.dprcファイル）を編集して行う。

GUIはシンプルで理解しやすいが、ワープロの編集機能がかつての日本語ワープロ専用機に似ていて、最近のワープロソフトに慣れたユーザーにはちょっととまどうところがあるかもしれない。また、一部スクロールバーの動作におかしなところがある。たとえば、スクロールバーを動かして文書先頭に移動させても、実際には文章の先頭部分が表示されず、カーソルキーでカーソルを上に動かすと表示が行われる。おそらく、実際のページサイズとスクロールバーに設定されている範囲がずれたのだと思われる。

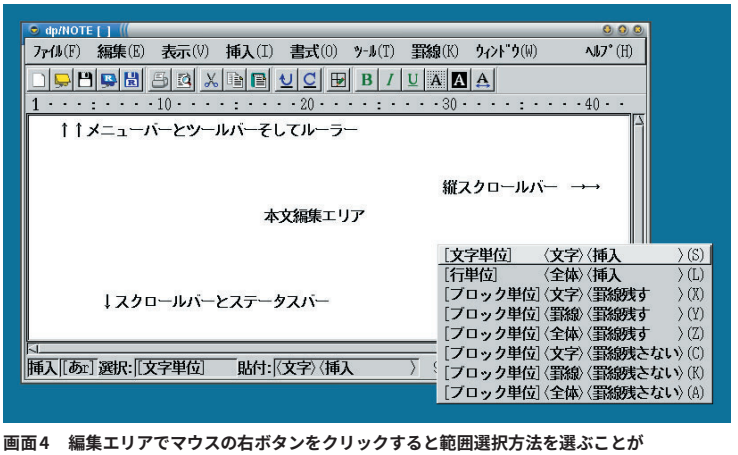
ワープロ基本機能

dp/NOTEでは、装飾付きのテキストのほかには表を配置できるのみで、グラフィックスファイルを配置することはできない。また、段組みの機能も持っていない。かつての日本語ワープロ専用機のようなものなのだが、最近のワープロソフトしか使ったことがないユーザーの方にはかえってわかりにくい表現かもしれない。たとえば、文字は任意にポイント数が指定できるのではなく、「横倍角」、「縦倍角」、「四倍角」、「横縮小」、「縦縮小」、「1/4縮小」という6つのバリエーションを使う（**画面5**）。「縦倍角」とは、文字を縦方向に2倍の大きさにするもので、横幅は変化しないため縦長の文字となる。標準の文字に対して、縦、横方向に2倍、1/2倍を別々に指定できるものなのである。かつてのワープロ専用機は、アウトラインフォントを持っておらず、ビットマップフォントを使っていたため、あまり大きく拡大するとギザギザの文字になってしまう。このとき、かろうじてドットのジャギーが目立たないのが縦横2倍程度だったわけである。また処理上、拡大縮小する文字の前にそれを意味するアトリビュート文字を置いて、テキストを管理していたため、画面上では特殊なアトリビュート文字と、本来の文字が並んで表示されていた。これは、初期のワープロ専用機の表示がたった1行のLCDだったり、ビットマップディスプレイではなかったなどの理由で、文字を指定されたサイズどおりに表示できなかったためでもある。

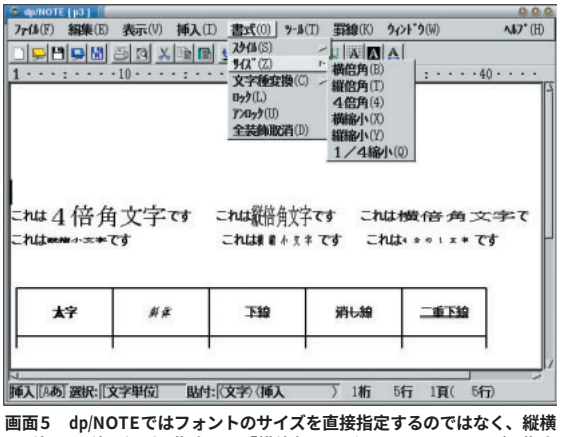
dp/NOTEでは、この文字サイズの指定とは別に、文字のスタイルとして「太字」や「斜体」、「下線」、「打ち消し線」、「二重下線」などが行えるほか、網掛けなどを含めて、全部で15種類の装飾が行える（**画面6**）。

また、文字のない部分にもカーソルを動かすことが可能な「フリーカーソル」となっており、なにもない場所にカーソルを置き、入力を行えば、そこから行頭にかけてスペースが入り、文字の後ろには改行が置かれる。また、該当行とそれ以前に入力した行の間が空白であれば、そこに自動的に改行が置かれる。簡単にいえば、ページ内の任意の位置にカーソルを置いて、いきなり入力が可能となる。パソコン系では、かつて一太郎が売り物にしていた機能と同じである。もっとも、国産の多くのワープロ専用機では普通の機能でもあった。

dp/NOTEはその開発時期が早かったこともあって、編集機能としては国産ワープロ専用機に近いイメージを持っているわけだ（当時の多くのパソコン用ワープロは、広く普及していたワープロ専用機をモデルにしたものが多かった。そのほうが、ユーザーに受け



画面4　編集エリアでマウスの右ボタンをクリックすると範囲選択方法を選ぶことができる。



画面5　dp/NOTEではフォントのサイズを直接指定するのではなく、縦横の2倍、1/2倍を個別に指定する「横倍角」などでフォントのサイズを指定する。

入れられやすいからである)。

ただし、dp/NOTEでの印刷は、PostScript経由になるため、VFlibを通して付属のTrueTypeフォントが使え、印刷された文字がギザギザになるようなことはない。あくまでもアプリケーションとしての機能がそうなっているだけである。

ワープロによくある「左寄せ」や「センタリング」は、段落の属性として設定するのではなく、文字の左側にスペースを置くことで行われるため、一度設定しても、編集後、再度センタリングなどを設定しなおす必要がある。

テキストなどの表示は、簡易モード(倍角指定文字の前に丸付き漢字で倍などとなるアトリビュート文字を置くもの。**画面7**)と、実際の表示に近い詳細モード(ただし、ページの区切りは表示されない)および、印刷プレビューがある(**画面8**)。

簡易モードは正確な表示にならない反面、動作が高速で長文の編集などに向いている。ただし、最近の高速なCPUならば詳細モードでも十分編集が可能だ。その意味では、dp/NOTEはかなり軽いソフトともいえるだろう。

ページ内に配置できる罫線は文字扱いで、同じ場所に文字と罫線を同時に配置することはできず、罫線の引かれている部分は文字が入力できなくなる。罫線は、罫線を描画するモードに設定し、カーソルキーを動かしつつ縦横の罫線位置をスペースキーで指定したのち、最後にEnterキーを押して描画させる。

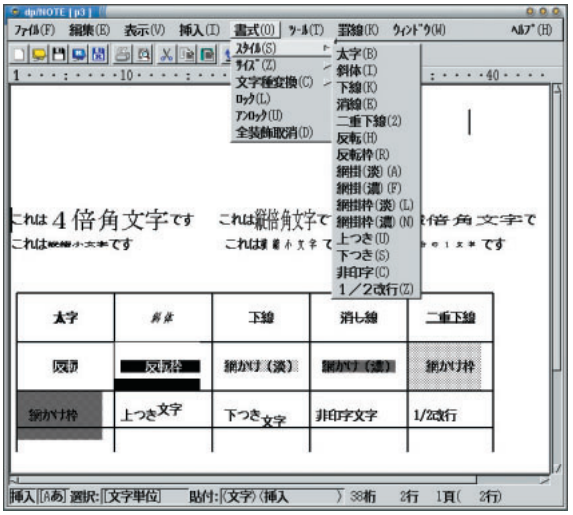
罫線は文字扱いであるため、罫線より上の行に文字が入力されると、罫線も下の行と一緒に動く。また、罫線の左側では入力された文字列は折り返しされる(**画面9**)。

罫線内に文字を入れることも可能だが、挿入の結果、罫線内からはみ出す場合、警告が表示されてはみ出した部分がカットされてしまう(**画面10**)。このあたり、自動的に矩形範囲を下に延長するような機能がほしいところだ。dp/NOTEでは、この罫線で囲まれた領域のサイズを変えるには、「桁挿入/削除」、「行挿入/削除」といった操作をメニューなどで指定して実行する必要がある。

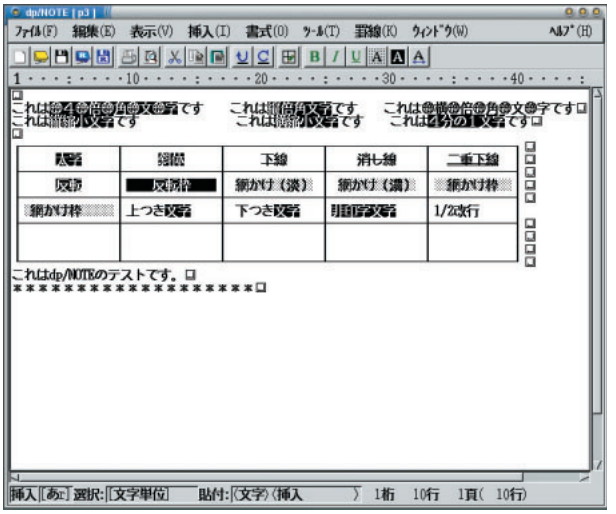
なお、罫線には、「実線」、「点線」、「二重線」など全部で8種類がある。日本では罫線を入れた文書を作成することが多く、単なる文書中の表だけでなく、たとえば全体が罫線で囲まれた、記入用紙のようなものも作成することがある。このような場合、英語版からの移植ソフトなどが持つ表作成という機能では不便なことがある。その点、dp/NOTEの罫線機能はページ全体を表にしてもあくまでも文字扱いとなるため、勝手に表が次のページに移動してしまうことなどがなく、使いやすいことは確かだ。

日本語の入力については、前述のようにWnn系のjserverおよび、kinput2経由でほかの変換エンジンを使うことができる。変換効率などは結局、外部のプログラムに依存する。なお、dp/NOTEは基本的にはjserverとの連携を想定しており、jserverを使う場合には日本語モード開始キーや、モード表示などが行える。また、jserverと直接交信している場合には、dp/NOTE内のメニューから単語登録を行うことも可能になる。

dp/NOTEのネイティブのファイルフォーマットでは、テキスト部分(拡張子なし)とアトリビュートファイル(拡張子.att)の2つのファイルが作られる。テキスト部分はEUCで記述されているようで、このままテキストとし



画面6　文字に対して、反転や網掛けなどのスタイルを適用することができる。



画面7　簡易表示モードは文字サイズが一定になり、一部のスタイルが省略されて表示される。

て扱うことができた。アトリビュート部分はバイナリ形式となっており、詳細は不明だ。dp/NOTEのファイルとして扱う場合には、この2つのファイルを常に一緒に扱う必要がある。dp/NOTEにはこのためか、2つのファイルを同時にコピー、移動するdpcpとdpmvコマンドが用意されている。これを使うと、拡張子を指定せずにdp/NOTEのファイルを指定すれば、cpやmvと同じような方法で2つのファイルのコピーや移動が行える。

HTML関連機能

dp/NOTEはHTMLを扱うことができる。これには、編集中にテキストとして直接タグを入力することを支援する機能（HTML編集。**画面11**）と、dp/NOTEのワープロとして作成した文書をHTMLに変換する外部プログラムから構成されている。

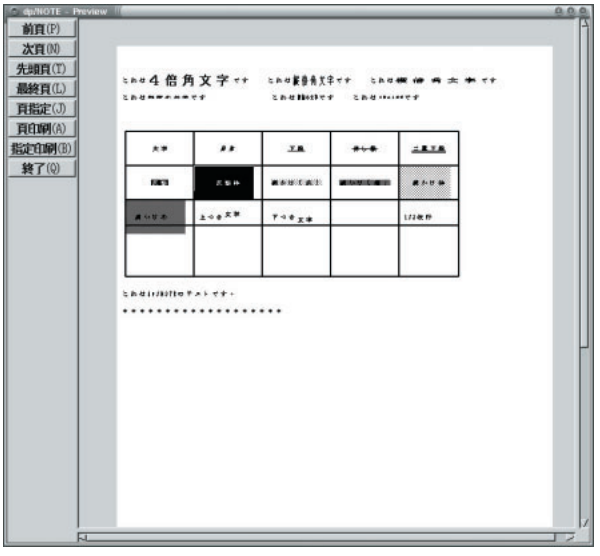
また、HTMLをPostScriptに変換することも可能になっている。これは、HTMLファイルを印刷するときなどに利用できる。

簡単にHTMLを作るには、dp/NOTEのワープロ機能で普通に作成した文書をHTMLに変換すればいいのだが、dp/NOTEでは画像を直接扱うことができないため、生成されるHTMLファイルも画像を含むことができない。逆に、タグを直接編集するHTML編集機能を使うことで、画像などを入れた文書ファイルを作り、これをPostScript化して印刷することができる。

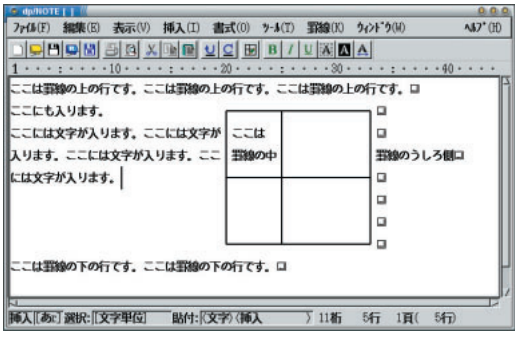
HTML関連の機能は、Webサーバなどに置くHTMLファイル作成用としても利用できるが、印刷までの機能を考えると、dp/NOTEでは表現できない文書を作成するための補助的な方法ともいえる。つまり、最初にdp/NOTEで文書を作成し、これをHTML化したのち、画像などを加え、PostScriptに変換して印刷するわけである。

なお、dp/NOTEの文書ファイルをHTML化する機能と、HTMLをPostScriptに変換する機能は独立したプログラムファイルとなっており、dp/NOTEのメニューから実行可能なほか、コマンドラインから直接実行させることもできる。このため、大量のファイルをバッチ処理的に扱うことも可能だ。

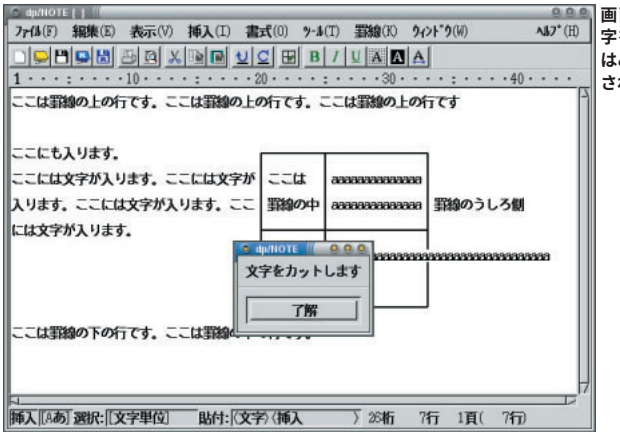
最近のUNIXやLinux向けのオフィスソフトは、発想がWindows的で、1



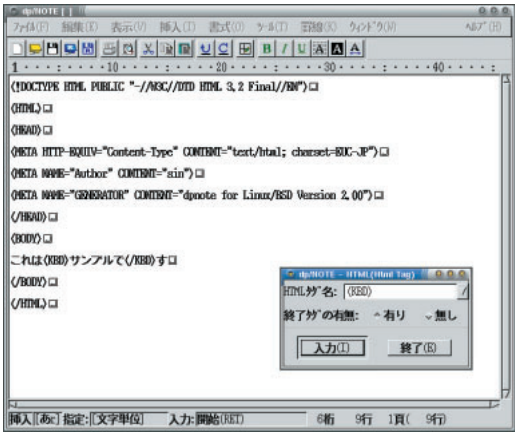
画面8　印刷プレビューではページごとのイメージを表示できる。なお、印刷はPostScript経由で行われる。



画面9　罫線を使って表などを作成できる。なお、これは罫線文字を使うものだが、罫線文字は文字入力から保護されるようになっており、文字挿入で罫線が壊れることはない。



画面10　罫線内に文字を入れる場合、枠をはみ出す文字はカットされてしまう。



画面11　dp/NOTEではHTMLファイルを直接編集する機能をサポートする。範囲を指定してタグで囲む機能や、テーブル用のタグを自動出力する機能がある。

つのプログラムに細かい機能をぜんぶ入れてしまって、外部プログラムを組み合わせて機能を実現するようなところがなく、こうしたファイルの変換機能だけを実行しようとすると困る場合がある。

たとえば、HTTPサーバ側で動くスクリプトで、最新の文書ファイルをHTML化して表示しようとした場合などである。アプリケーションにとって、さまざまな情報を最も保持できるのはネイティブのファイル形式で、HTML化することなどにより各種の情報が失われることがある。このようなとき、ファイルの管理はネイティブ形式にしておき、必要に応じてHTMLに変換できると、常に最新の情報を表示できる。

このような場合には、ファイルの変換機能だけを実行できる別プログラム形式が便利なのである。UNIX系では、小さなツールを組み合わせて機能を実現するといったことが普通に行われている。これによってカスタマイズしたツールを作ることができる。ある意味、dp/NOTEでファイル変換機能が別プログラムとなっているのは、この手法に沿ったものといえるだろう。

電子メール

dp/NOTEは電子メールを扱う機能を持っている。dp/NOTEのメール機能は、外部コマンド（dpbiff）を使って、POPサーバからシステムのメールスプール（/var/spool/mail/＜ユーザー名＞）にメールを読み込み、そのあと、各ユーザーのホームディレクトリにメールを保存する。このメール機能は、フォルダ管理、メッセージリスト、本文表示を1つのウィンドウにまとめたものだ（**画面12**）。

dpbiffはいわゆるbiffプログラム（**画面13**）で、別プログラムとなっており、一定時間ごとにPOPサーバやメールスプールをチェックして新着メールの有無を調べるものだ。POPサーバを使う場合、このdpbiffが、POPサーバにあるメッセージをローカルのメールスプール（ユーザーの指定する任意のディレクトリ）に取得することができる。

送信に関しては、sendmailを直接起動するほか、メールの取得用コマンドとあわせて、ユーザーが任意のコマンドを指定することもできる。

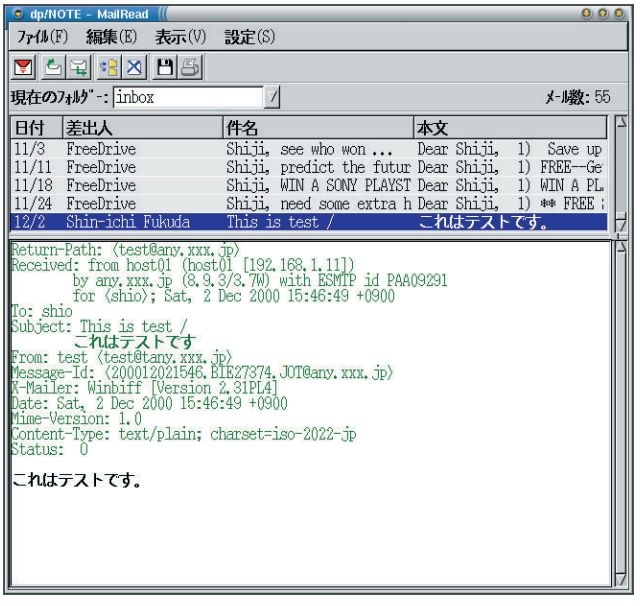
ただし、メール機能としては基本的なものしかなく、本文はMIMEなどには対応しておらず（ヘッダ内のSubjectなどのBase64エンコードには対応している）、メッセージ本文は単なるテキストとしてしか扱うことができない。

また、送信メッセージはdp/NOTEで編集している文書か、すでに保存されている文書を利用する。送信時には文書をテキスト化し、指定されたコマンド（通常はsendmail）を使ってメッセージを送信する。

外字機能

dp/NOTEはいわゆる外字を作成して文書中で利用する機能を持っている（**画面14**）。外字とは、字体をユーザーが定義して利用するもので、コードとしては外字領域（JISコードの7F21以降）を使うのが普通だ。

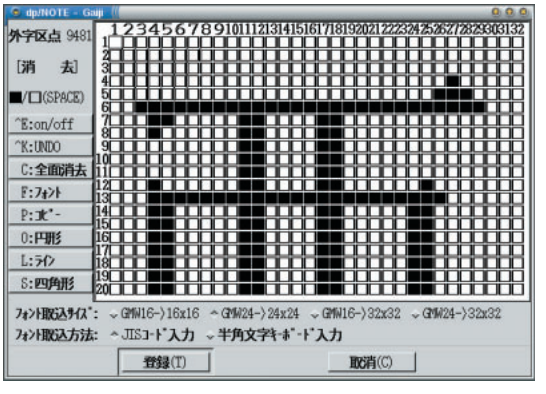
dp/NOTEでは、外字エディタを使って32×32ドットの外字をユーザーが作成可能で、作成した外字は個人ごとに管理される。外字は、JISコードにない文字を表示、印刷するためのもので、基本的には外字コードによるデータ交



画面12　dp/NOTEにはメール機能が組み込まれており、dp/NOTEで直接メールを受信できる。



画面13　dp/NOTE付属のdpbiff。これは、POPサーバからメール着信の有無を調べて通知するだけなく、メール取得機能も含んでいる。



画面14　dp/NOTEでは、32×32ドットの外字を定義して利用することができる。外字は専用の外字エディタで作成が可能だ。

換は想定されていない。相手のシステムでは、同じ字体を定義していない限り、画面には違ったものが表示される、あるいは表示が不可能となるからだ。しかし文書を印刷するうえで、JISで定義されていない人名文字などが必要となることは多く、以前からこれを外字という仕組みでカバーしていたわけだ。

dp/NOTEで使用される外字領域は、JIS第二水準の最後にある未使用領域のコード（7E7E以下）が使われる。

この外字エディタでは、既存のフォントからのコピーと、ドットのオン／オフ、円、矩形、線などの描画を行うことができる。既存フォントからのコピーは、JISにない異字体などを作るときに利用でき、図形は、たとえばマークのようなものを作るときに使う。

外字データは、各ユーザーのホームディレクトリにあるDp/gaijiディレクトリの3232.gimgファイルに記録されており、外字パターンは、ASCIIアートのようにスペースと「#」を使って記録されている（**画面15**）。

Microsoft Wordとの文書交換

dp/NOTEは、Microsoft Wordとの文書交換をRTF（Rich Text Format）形式で行う機能がある。RTF形式とは、テキストだけを使って文字や段落属性などを表現したもので、もともとは、DOSとMacintosh用のMicrosoft Wordで文書交換を行うためのファイル形式であった。しかし、テキストだけを使った形式であるため、ファイルの転送が容易で、他社製ソフトウェアなどでも広く対応しているため、交換用フォーマットとして比較的よく使われている。ただし、このRTF形式ではファイルを生成するMicrosoft Wordの機能に依存する部分があり、本文テキストが失われることはないが、必ずしもレイアウト情報を完全に受け渡しできるわけではない。また、Microsoft Wordの設定やバージョンにより、記述方法（というか同じ書式の表現方法）が違うこともある。

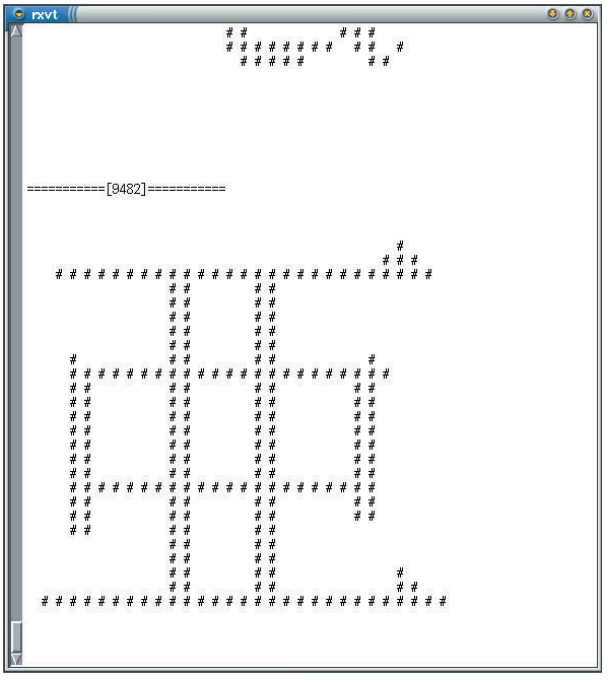
dp/NOTEは、Microsoft Word 97相当のRTF形式ファイルの読み書きが可能になっている。機能としてはdp/NOTEから直接利用できるが、RTFファイルをdp/NOTEの形式に変換する機能も別プログラム（rft2dp）として実装されている。これを使うことで、シェルスクリプトなどを使って大量のRTFファイルをまとめて変換することも可能だ。

雑感

dp/NOTEは、最近登場したソフトに比べると、開発時期が早かったせいかワープロの機能としては古いものを感じる。特に文字サイズをポイントなどで自由に指定できないあたりは、実用上の問題は少ないにしても、最近のユーザーから見ればかなり「特異」な仕様に見えるだろう。

また、手作業による設定項目が多いのも、Emacsのように古くからのUNIXソフトウェアらしさといえないこともないが、GTK+などを使うソフトウェアなら、GUIベースでの設定機能がほしいところだ。

しかし、開発時期が早かったことは、それだけアプリケーションとして「枯れて」いるともいえ、安定感があるというのは見逃せない部分でもある。立派なことに、今回の評価中ではdp/NOTEが落ちたり、反応がなくなるということは一度もなかった。Linux上のOfficeアプリケーションは増えてきてはいるが、この点ではまだまだというものが多い。どのソフトウェアでも安定感は必要だが、文書作成など、落ちるとやり直しのきかない作業を行うものは特に重要である。このような状態になるまでには、基本的には時間が必要で、先行して開発されたdp/NOTEには、そうした強みがあるわけだ。



画面15　作成された外字は、テキストファイル（3232.gimg）にスペースと「#」でASCIIアートのように記録されている。

隠喩としてのコンピュータ

デビッド・チャウムのeキャッシュ再考

文：豊福 剛
Text : Tsuyoshi Toyofuku

このところ国内でLETSを立ち上げる動きが活発になってきているようだ。LETSを運用するためのソフトウェアをオープンソースで開発しているプロジェクトも立ち上がっている。

中でも鈴木健氏が中心になって進めているGETS（Glocal Exchange Trading System）は、Perl＋DBMS（PostgreSQLなど）による構成のプロトタイプが完成し、このPerl版に加えてJava版の開発も計画されている意欲的なプロジェクトだ（http://sacral.c.u-tokyo.ac.jp/~ken/gets/）。

また npo-cap@egroups.co.jp（NPOと資本主義の新しい関係を探るML）では、地域通貨、協同組合、NPO、フリーソフトウェア／オープンソース、P2Pなど、より包括的なテーマ群をめぐって活発な意見交換が展開されている。興味のある方は、一度アクセスされてみてはいかがだろうか。

スマートカードとLETS

『広告』という博報堂が発行している隔月刊の雑誌は、たとえば2000年9＋10月号では、2ちゃんねるを特集したり、ノートルダム女子大のよしだともこを取り上げていたりしたので、そこそこ注目していたら、2000年11＋12月号では、地域通貨LETSの創設者であるマイケル・リントンが写真つきで紹介されていた。興味深いことに、スマートカードを利用したLETSも展開されているようだ。イギリスのGIS社（http://www.gis.co.uk/）が製造した電卓みたいな形状のデバイス（トランザクタと呼ばれている）の写真、レストランのレジなどに設置されているスマートカードリーダの写真などが掲載されている。レストランなどでの清算の一部をスマートカードに記録されたLETSで支払うことができるようだ。

残念ながら2ページほどの短い記事なので、これらのデバイスがどのように実装されているのかはわからないのだが、GIS社の製品紹介ページを見ると、スマートカード型電子マネーとして有名なモンデックス用のデバイスもあるみたいなので、技術的にはモンデックスとかなり共通する部分があると思われる。スマートカードには、LETSの口座残高から引き落とした額が充填されるのだろうし、スマートカード間でLETSを移転させるときには、トランザクタやカードリーダ・ライタを使うのだろう。

また、携帯電話を利用したLETSも企画されているようで、ノキア社に接触をはかっているらしい。トランザクタ

にはオプションとしてモデムも提供されているくらいだから、電話機にスマートカードを挿入する方式なのだろうか。

ともかく、スマートカードの応用事例は電子マネーだけに限定されないのであり、スマートカードを前提とするシステムを、一方的に押し付けられるのではなく、自前で構築することが可能なことに、ちょっと驚いたのだ。暗号処理専用プロセッサが組み込まれているとはいえ、スマートカードがCPUとメモリを持つコンピュータとして構成されているのであれば、我々が使いたい方法で使える技術であってほしい。

LETSで決済するときは、パソコンに向かって操作するしかないとしたら、LETSで取引されるサービスやモノの種類は、著しく制限されてしまうだろう。LETSを日常生活の中に違和感なく浸透させるためには、スマートカードのような道具が必要なのだ。

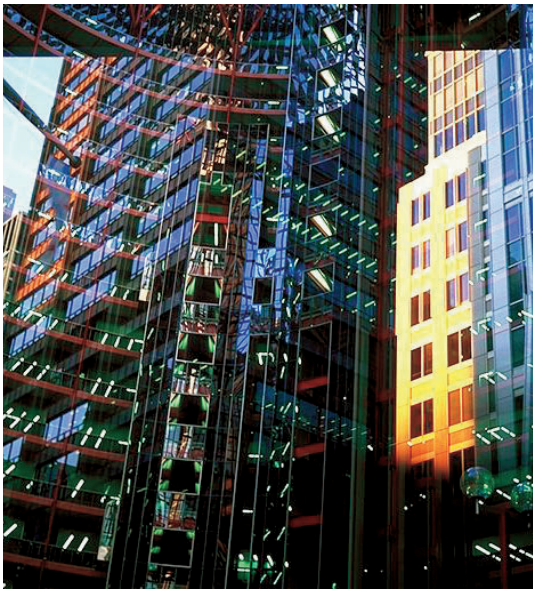
eキャッシュと匿名性

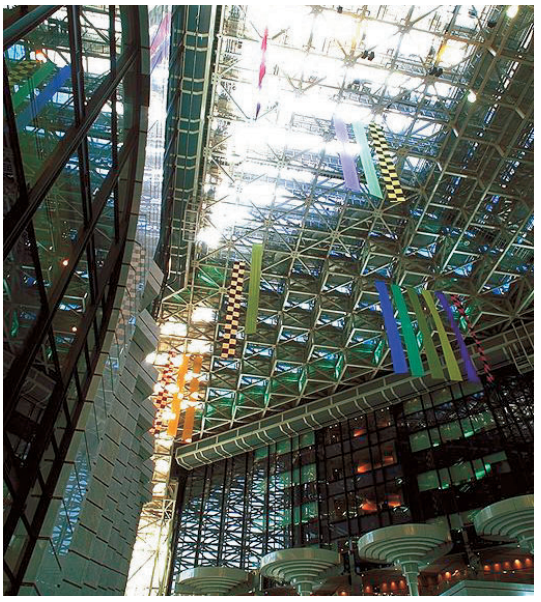
ところで、電子マネーにはスマートマネー型のほかに、ネットワーク型と呼ばれるタイプのものがあった。電子マネーの実現に必要とされるデータのチェック機能、保管機能、不正アクセスや二重使用の排除などを、スマートカードを使わずに、ソフトウェアだけで実現しようとするのが、ネットワーク型電子マネーのアプローチだ。このネットワーク型の電子マネーの考案者として有名なのが、デビッド・チャウム（http://www.chaum.com/）だ。彼はオランダのデジキャッシュ社のプロジェクトに参加し、eキャッシュというネットワーク型電子マネーを開発した。

デジキャッシュ社のeキャッシュは1995年頃に非常に話題になったものの、その後デジキャッシュ社が倒産したこともあって、ネットワーク型電子マネーはスマートカード型に比べると、あまり普及しなかったようだ。しかし、チャウムのアイデアは、匿名性を重視する思想が背景にあるため、これを失敗した技術として忘れてしまうのではなく、LETSのような帳簿を公開するアプローチ（そこでは非匿名性が前提になる）と比較検討することは、意味があるだろう。

eキャッシュの仕組みを単純化して説明すると、次のようになる。

まず利用者はeキャッシュ用のソフトウェアを入手し、仮想銀行に口座を開設する。仮想銀行からeキャッシュを発行してもらいたいときは、要求金額を伝え、その要求金額と発行番号を記した文書を暗号化したものを仮想銀行に





送る。仮想銀行は利用者の要求金額分を減算し、暗号化されている文書にデジタル署名（仮想銀行の秘密鍵を使う）を付加したものを利用者に送り返す。

eキャッシュによる決済は次のようになる。利用者Aは利用者Bにインターネット経由でeキャッシュを送金する。利用者Bは、仮想銀行の公開鍵を使って、eキャッシュのデジタル署名をチェックする。利用者Bは受け取ったeキャッシュを仮想銀行に送る。仮想銀行では、そのeキャッシュが二重使用されたものでないかチェックし、問題がなければ、利用者Bの口座に金額分を加算する。

AからBへeキャッシュが移動したことが、仮想銀行に対して匿名であると保証されるのはなぜか。仮想銀行は、Aにeキャッシュを発行し、Bからeキャッシュを受け取っているが、そのeキャッシュがAからBに渡されたものであることを確認できないのはなぜか。

仮想銀行が発行するeキャッシュは、利用者によって暗号化された文書に仮想銀行がデジタル署名を付加したものであった。実はこのデジタル署名には、ブラインド署名と呼ばれる特殊な手法が使われている。利用者Aが仮想銀行に送る文書は暗号化されているので、つまり文書を封筒に入れた状態で送っていると考えればいい。このため仮想銀行は封筒の中身を確認することができない。そして、封筒は、実は裏側にカーボン紙が付いている封筒になっている。仮想銀行が封筒に署名するとき、その署名は封筒の中の文書にも転写される。利用者Aは、封筒から中身の文書を取り出して、その文書を利用者Bに送る。この文書には金額と発行番号が記されているが、発行番号は利用者Aが発行したものであり、仮想銀行にはそれを発行したのが利用者Aであることはわからないのだ。

封筒への署名が封筒の中の文書に転写されると説明したが、より正確には、暗号化された文書Zに対するデジタル署名（ブラインド署名）Qをもとに、暗号化されていない文書Xに対するデジタル署名Mを求める、ということである。ちょっと不思議な感じもするが、これが可能であることは、数学的に証明されている。

不正使用監視の方法

eキャッシュが改ざんされていないかどうかは、仮想銀行によるデジタル署名によって確認できる。それでは、eキャッシュの二重使用を防ぐための方法論はどうなっているのだろうか。いったん発行されたeキャッシュを、実際に使用する前にコピーしておいたとしたら、そのコピー

（仮想銀行によるデジタル署名を含む）を何度でも使用できてしまうのではないだろうか。

このような二重使用を防ぐためには、たとえば仮想銀行がeキャッシュを発行する際に、ある一意的なIDをeキャッシュに持たせておけばいい。しかし、そのようなIDがあれば、そのeキャッシュがどのような経路で利用されたのか追跡できてしまうので、eキャッシュの匿名性は失われる。

eキャッシュには発行番号が書き込まれているので、この発行番号が重複していれば、二重使用されたことがわかる。しかし、発行番号は、仮想銀行ではなく、利用者が発行したものであるため、二重使用した利用者を特定することはできない。

eキャッシュは、この問題を解決するために、独特の面白い方法を使っている。岩村 充『電子マネー入門』（日経文庫）によると、「電子マネーのユーザーが不正にマネーを使用しない限りはパーソナル情報を誰にも見せないが、いったんマネーをコピーして二重使用したりすると、そうした不正を働いたユーザーの名前が明らかになってしまうような仕掛け」が施されているらしい。

実は仮想銀行から発行されるeキャッシュは、必ず2つのデータがペアになって発行されている。この2つのデータには、eキャッシュの発行を受けたユーザー名が分割されて書き込まれている。仮にユーザー名が「ナツメソウセキ」だとすると、一方には「ナメウキ」、他方には「ツソセ」が書き込まれているとする。eキャッシュを誰かに送金するときに、このペアになっているデータのどちらを使うかは、プログラムによってランダムに選択される。

もし、コピーが使用された場合、50％の確率で、仮想銀行に「ナメウキ」データと「ツソセ」データが還流することになるので、「ナツメソウセキ」が二重使用者であることが判明する。とはいえ、50％の確率でしか判明しないともいえるわけだが、たとえば1000円分のeキャッシュが発行されるときは、それは百円玉データが10個という小銭データとして発行されるので、ユーザー名の割り出しから逃れるためには、10個のデータそれぞれに対して、プログラムによって選択されたデータと同じものを選ばなければならない。それが成功する確率は0.5の10乗になるというわけだ。

eキャッシュにおける匿名性の確保と不正使用の監視機能は、たいへん興味深い。利用者の口座残高がゼロであってもeキャッシュを発行できるようにすれば、それは取引情報がサーバに残らないLETSになるのではないだろうか。

Profile

とよふく つよし

1962年生まれ。メディアデザイン研究所技術顧問。訳書に『Javaプログラムクイックリファレンス』『Java分散コンピューティング』（オライリージャパン）『GIMPパーフェクトガイド』（エムディーエヌコーポレーション）などがある。

モダンジャズとソフトウェアの革命家たち

文：安田幸弘
Text : Yukihiro Yasuda

「バード」の想い出

チャーリー・パーカーという天才アルトサックス奏者がいた。もう50年も前に死んでしまった昔の人なのだけれど、数カ月前、都内某所で「チャーリー・パーカー生誕80年記念ライブ」が開かれた。

チャーリー・パーカーと言っても、ほとんどの人は「そんな名前は聞いたこともないね」だろうし、モダンジャズが好きな人でも、「パーカー……古いねえ」という反応のほうが多いぐらいだ。あるいは映画ファンなら、彼を主人公にした「バード」という映画があったことを覚えているかもしれないが、その程度のものだろう。そんなわけで、どうせ会場はガラガラだろうとたかをくくってゆっくり出かけたら、小さなライブハウスは、ぎっしりと埋まっていて、あやうく立ち見になるところだった。

前半はトークセッションとやらで、高名なジャズ評論家が垂れ流す御託はあんまり面白くなかったけれど、生前のパーカーと親交があったというトニー・スコットというクラリネット奏者が登場すると、何やらオモシロゲな雰囲気になってきた。彼もすでに80歳近い高齢で、ヨレヨレのおじいさんで、パーカーの思い出話もヨレヨレなんだけれど、話の合間にクラリネットを構えると、それまでのヨレヨレ加減がウソみたいにシャキッとするというあたりは、さすがに名人である。満場のパーカーのマニアと当時の記憶に触発されたのか、その後の日本のプレイヤーとのセッションもすばらしいものだった。

即興演奏の天才

ところで、このチャーリー・パーカーというサックス奏者、専門家に言わせれば、バッハやモーツァルトに匹敵する天才音楽家なんだそうだ。その割には全然有名じゃないのは、パーカーの音楽の真髄が徹底的に即興演奏にあったということ、親しみやすいメロディーを残さなかったということなどが理由だろう。また、パーカーが白人には理解できないようなエキセントリックなサウンドを意図的に吹いていたという説もある。

とにかくチャーリー・パーカーは、その場の即興で、誰も聞いたことがないような、とんでもなく複雑で魅力的なサウンドを次々と生み出していった。そして彼が即興で生み出すサウンドの構成やフレージングは、1950年代以後のジャズはもちろん、ロックやポップスを含めて現代のあらゆる音楽に何らかの形で直接的、間接的に影響を与えた大天才なのである。少なくとも、記念ライブの参席者たちは異口同音にそう言っていたし、ぼくもそう信じているわけなんだけど。

フレーズには著作権がない？

大天才と呼ばれる人物には、逸話がつきものである。当然、かなり破天荒な人物だったらしいチャーリー・パーカーにも、数多くの逸話やエピソードが残されている。女グセが悪かったとか、突然大金を持って豪遊していたかと思うと次の日には一文無しになって借金をしたとか、まあいろいろあるけれど、そんなエピソードのひとつに、彼が何かでレコード会社のディレクターと支払いに関してもめたとき、「なんでこんな曲に著作権料なんか払うんだ？ 誰もがオレのフレーズをコピーして演奏しているのに、なんでオレには著作権料が入ってこないんだ？」と食ってかかったという話がある。

そのディレクターが何と答えたのかは記録に残っていないが、当時のサックス奏者は誰もが彼のフレーズを使って演奏していたというのだから、もっともな話ではある。昔、この話を聞いたときに、「そうだよなあ」と思ったのだけれど、今になって考えてみると、たぶん実務的には楽譜やレコードのように、形になっていないものは著作権で保護されず、理屈としては部分的なフレーズの

引用や音の組み立て方といった一般的な理論も著作権では保護できないということなのだろう。もし、フレーズに著作権が認められれば、おそらくビル・ゲイツなんか足下にも及ばない大金持ちになっていたに違いないが、パーカー自身は著作権料なんかよりも、「みんなオレのフレーズを吹いてるぜ」ということに満足していたともいう。

とはいえ、パーカーを含めて当時のジャズプレイヤーたちは、著作権の問題を回避するためにちょっとした手を使うのが常だった。そもそも即興演奏が主体のモダンジャズに、原曲のメロディはたいして重要じゃない。曲を構成するコード進行と呼ばれる和音のパターンに従って即興演奏が進められるのだが、幸いなことにコード進行もまた著作権の保護を受けない部分だった。そこで原曲のコード進行だけを借りて適当にメロディを作り、著作権料の問題を避けていたわけだ。

ただし既存のコード進行を借りると言っても、モダンジャズを創始したチャーリー・パーカーの仲間たちは、単に原曲のコード進行をそのままなぞって満足していたわけではなかった。彼らは毎晩ニューヨークのジャズクラブに集まり、原曲のコード進行に複雑な変更を加え、複雑怪奇な和音を次々と開発していっただけでなく、リズムやメロディという面でも、「誰が一番変わったサウンドを出すか」を競いあっていたという。こうして、モダンジャズという音楽の方法論的な革命が進んでいったわけだ。どんなにパーカーが天才だったとしても、このような場がなければ、彼一人でこの革命は不可能だっただろう。

そして、新しい音楽の実験室のような深夜のジャズクラブのアフターアワーセッションに、著作権だのビジネスだのが入り込む余地なんて、まるでなかったに違いない。フレーズを真似されれば、さらに高度な、誰にも真似のできないフレーズを考えるだけのこと。そこではどんなに有名なプレイヤーも実力がなければ笑い物にされて逃げ出さざるをえなかったという。既存の古くさい音楽に満足できなかったジャズの革命家たちの中で、次々と新しいアイデアを出し、斬新なフレーズを作り出すことができるプレイヤーだけが、注目と尊敬を集めたのである。

名誉とお金

深夜の「記念ライブ」の会場で、ぼくはふとエリック・レイモンドが、フリーソフトの開発者は金ではなく名誉、あるいは自身のエゴを充足させることを重んじると喝破したことを思い出した。確かにその通りなんだろう。ただ、レイモンドは、フリーソフトウェアの開発の手法をソフトウェア開発の新しいマネジメントの方法論として注目したわけだが、単に「ブタもおだてりゃ木に登る」ってもんでもないだろうとも思う。

フリーソフトウェアは、半世紀前のモダンジャズの革命のように、商品としての既存のソフトウェアに満足できない人々の間で持ち上がった文化的なムーブメントとしての側面を持っている。そして、このムーブメントの活動を支える前提が、ちょうどモダンジャズの世界で、コード進行やフレーズの借用のような、ソースコードの共有なのだろう。

もちろん、どんなに尊敬とエゴの満足を与えられても、それだけでは飯は食えない。著作権に噛みついたパーカーも、レコードの印税が大きな収入源だった。ビジネスの世界にはルールが必要で、ソフトの世界でも著作権や知的所有権というやつが有効に機能しなければならない。ただ、知識や情報に関するルールは、現在のように商品としての情報を保護するものとしてばかりでなく、人類の共有財産としての知識を共有し、その発展を保証するものでなきゃいけないと思うのである。

Profile

やすだ ゆきひろ

生業はテクニカルライター。原稿書きのかたわら、(株)市民電子情報網のボランティア社長兼技術スタッフとして、NGO向けプロバイダのネットワーク運営に携わる。

ドクターShiodaの

ギョーカイSnapShot

PC業界今月の話題

文：塩田紳二
Text : Shinji Shioda

11・30 NTTドコモ、AT&T Wirelessに出資
11・30 楽天、InfoSeek日本法人買収
11・30 Crusoeの一部ロットに欠陥。NECは交換を発表
11・24 ROBODEX2000開催
11・21 Pentium 4発表
11・17 ICANN新TLDを承認

休みが来たらゆっくりと、と思うことはいっぱいあるのだが、なかなかその休みが来ない。おそらく2001年も、忙しいはずである。

ビル・ゲイツ、PCの重要性を説く

このところ大きなニュースのないMicrosoftだが、11月に開かれたCOMDEXでビル・ゲイツは、基調講演でパソコンの重要性が高まるという趣旨の話をした。結局“.NET”の話なわけだが、高性能のサーバにデータを置き、高性能のデスクトップで分析、処理するという両方が高性能という対称的な仕組みを考えているらしい。これは、両者に性能差のあるサーバ・クライアントでもなく、どれもがクライアントであるピアツーピアでもないという。ゲイツは、これを「ソフトウェアツーソフトウェア」と呼んだそうだが、なんだかさらにわからない呼び方である。

また、キーボードのない「Tablet PC」のデモも行ったが、これはCrusoeのWeb Padによく似ている。違うのは、Tablet PCは次期Windowsである Whistlerで動作していることだ。

Microsoftがこういうハードウェアを含めたデモをするのは、気になる他社の動きがあって、それに対抗するときだ。かつて、ペンコンピュータが流行った（といってもほとんど製品が出なかったが）とき、Microsoftも急遽Windows for Pen Computingという製品をアナウンスした。また、同時にWindowsをOEMするハードウェアメーカーを集め、これを支持させ、業界を2つに割るような動きに出た

このTablet PCもおそらくこの分野で急成長中のLinuxを意識したものだ。Crusoeを採用しているのは、Microsoftの仕様を採用するであろう企業をバックに、Transmetaに圧力をかけるつもりなのだ。また、こうした機器を計画しているハードメーカーに対して、Linuxだけでなく、Whistlerもありますよという選択肢を提案することで、Linuxの採用を押さえるつもりなのであろう。

また、Palmとの戦いで不利なWindows CE（Windows Poweredだっけ？）を、高性能にするのは限界があるとして、ノートパソコンよりも手軽に使える分野（かつてCE Proが受け持っていたあたり）に、Windowsそのものであるwhistlerを持ち込み、PDA系に対して上から押さえにかかるという意味もあるのだろう。

PalmはARM系CPUへの移行がウワサされており、そうなると現在のCEを採用したPocket PCやハンドヘルドPCと性能的にほとんど違いがなくなってしまう。Pocket PCなどは現状でさえ不利なため、もっと不利になる。これだけならいいが、さらにPalmが上を目指すとすると、ノートパソコンなどを含めた、モバイル市場をかなり浸食される可能性がある。そこで、Pocket PCの親玉みたいなTablet PCが押さえとして必要になってくるわけである。

ただ、Tablet PCの登場は、2002年というかなり先の話。Microsoftの時計からすれば、どうなっているかわからないぐらい未来の話だ。

インテルようやくPentium 4を出荷するも前途多難

インテルは11月にようやくPentium 4の正式出荷を発表した。しかし、この発表、ウワサによると10月末が予定されていたものの、PCIビデオカード関連のトラブルで、発表が延期されたという。また、このPentium 4出荷直後にBIOSにバグが見つかり、一部の製

品は店頭でアップグレードすることを要求されたという。

ただ、修正はBIOSなのだが、バグがあったのはCPU自体という話があった。実はインテルのCPUには、CPUのマイクロコードをあとから修正するための機能が組み込まれており、これはBIOSが起動時にCPUに送り込む。一見、BIOSのプログラム自体にバクがあったように見えるが、実際にはその中にはCPUに対するパッチが含まれていることがあるのだ。各メーカーが一斉にBIOSをバージョンアップするなんてのは、実際には、CPUのパッチが含まれている可能性が高い。

さて、このPentium 4だが、どうもAthlonとのベンチマークで分が悪いようである。もっとも、これは最初からわかっていたこと。Pentium 4はクロック周波数を高くできることを前提に設計されており、同じクロック周波数だとすると、Pentium IIIよりも遅くなっているからである。

あちこちのベンチマークだと、1.5GHzのPentium 4は、1.2GHzのAhtlonを完全に上回ることはなく、プログラムによっては、負けることもあるらしい。個々のベンチマークに優劣をつけなければ、1.5GHzのPentium 4がおそらくAthlonの1.2GHzに相当するということなのだろう。

というわけで、2001年もクロック競争が熾烈になることは間違いない。Pentium 4は、ひたすらクロックを上げるために生まれたようなものなのだから。2001年中には2GHzに達するというのだから、2001年末にはどれぐらいになっているやら。しかし、値段が一緒ならクロックが高いほうがいいなんてことは競争があるから言えるのであって、これが1社独占なら、べらぼうな値段だったに違いない。

インテルには、もう1人ライバルがいる。それはTransmetaである。このTransmetaだが、NECやソニーなどが採用した一部ロットのCrusoeに不具合があったらしい。NECはさっそく交換ということになった。出だしとしてはちょっとまずいトラブルだが、一部ロットということで、全体的な問題ではないところが救いか。

ThinkPad 240にCrusoeを載せた試作機を作ったIBMは、とりあえずCrusoe採用マシンの計画をとりやめたらしい。消費電力が思ったほど下がらないというのが理由だが、裏にはいろいろあるような感じである。インテルとすれば、一度試作機まで作ったIBMをひっくり返したということで、ある程度の実績をアピールできるだろうし、まだ、動いていない米国系のノートPCメーカーに対する売り込みもしやすくなる。それに、Crusoeは、どこあろうIBMの半導体ラインで作られているのである。つまり、（実際にはどうだかわからないが）外から見れば、Crusoeを熟知しているはずのIBMが断念したというのだから、インテルの超低電圧版のほうがいいと受け取られる可能性は高い。

IBMは、Crusoeのおかげでインテルが低消費電力という方向に向かってくれたなんて発言をしていたが、逆にIBMは、インテルに対する圧力としてCrusoe試作機を作ったのかもしれない。

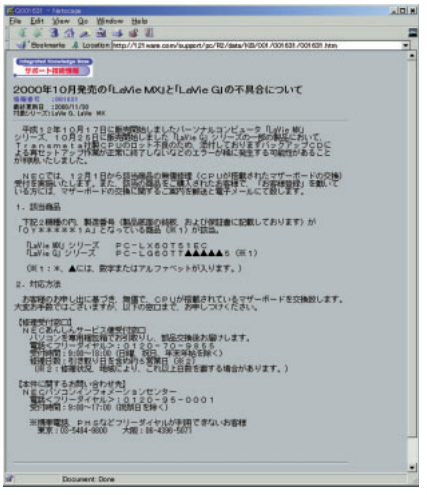
さてLinux業界は？

リーナス・トーバルズは2001年5月に自伝を出版するらしい。自分に関する本についてはこれが初めてとなる。

さて、日本では、「中華の鉄人」がTurbo Linuxを採用したというニュースが流れたが、これは、同氏が顧問を務める宅配中華料理サイトの構築にTurbo Linuxが使われたということらしい。最近はロボットばやりなので、Turbo Linuxで動く中華料理ロボットかと思ったが、そうではないみたい。ところで、いま米国では同氏を有名にした料理の鉄人が放送されて人気なのだとか。こういうニュースは、米国では配信されるのかしらね。TurboLinuxのWebサイト（日本、アメリカ、そして中国！）には、そういったリリースはどこにも載っていないようだが……。



COMDEXでのビル・ゲイツへのインタビュー
http://www.key3media.com/comdex/fall2000/daily/keynotes/gates_interview.html



NEC Crusoe搭載ノートパソコンの一部をCPU交換
http://121ware.com/support/pc/R2/data/KB/001/001631/001631.htm

日刊アスキーLinux on Linux magazine

http://www.linux24.com/

日刊アスキー Linuxの舞台裏

～2001年のLinuxアンケート～



本稿執筆時はまだ21世紀に突入していないが、日刊アスキー Linuxでは年末年始の特別企画として、「2001年のLinux」と題し、Linux関連企業や団体へのアンケートを行っている。主旨としては、来年のLinuxはどうなるか、そのときの戦略はというもので、各社から回答をいただいた。全文はWebのほうをご覧いただくとして、ここでは現時点で集まっているコメントの中から、ディストリビュータ関連のものを要約、抜粋してお届けしよう。

（吉川大郎）

レッドハット

まず、トップバッターはレッドハット。同社代表取締役社長・平野正信氏のコメントだ。新しく提携したVine Linuxにも触れている。

「インターネットサーバだけではなく、基幹業務に欠かせないERP、CRMなどの分野で利用されると思われる。実際、日本アイ・ビー・エムなどによって1万台以上の導入事例などが出てきている。同時に、デスクトップ分野でも日本語環境に対応したオフィスソフトのリリースが予想され、この分野のユーザーも増えていくだろう。当社は基幹業務にはRed Hat Linux、デスクトップ分野ではVine Linuxとユーザニーズに応じた製品を用意すると同時に、シームレスなサポート体制を敷くことで、幅広い対応を行っていく」

「サービスの枠を広げ、インターネットを介したLinuxサポートビジネスを開始する。具体的には、北米ではRed Hat Linux7のリリースと同時に試験運用されている『Red Hat Network』（サポート情報やアップデートをインターネット経由でプロアクティブに提供するサービス）を国内でも開始する。このサービスを核に、インターネットインフラのスタンダードを確立し、オープンソースベースの企業システムおよび組み込み製品の普及、ならびにより多くのアプリケーションサービスの参入を促していく」

ターボリナックス ジャパン

次は、ターボリナックス ジャパン代表取締役社長・小島國照氏だ。各パートナー企業との連携を強化すると語る氏は同時に、カーネル2.4のディストリビューション実装タイミングについても触れている。

「ターボリナックス ジャパンの基本戦略は、今後も引き続きLinuxディストリビューションの製品強化と、サポート・サービス体制の拡大である。

Linux市場を拡大するためには、ディストリビューション・パッケージの販売強化に加えて、Linuxを使いこなすためにサポートやコンサルティングといったサービスの充実が不可欠と考え、パートナー企業との連携により、サポート・サービス体制の充実、Linuxエンジニアの育成を実現していく。

また、開発・エンジニア部門を中心として、人員の増強も含めて、社内体制をより一層強化していく」

「カーネル2.4をいつディストリビューションで正式採用するかは2001年初頭の焦点になるが、当社が製品として出す以上、企業システムで安心して使えるLinuxディストリビューションに仕上げてから提供する。そのためTurbo Linux Workstationの新版（V.7）では不安定なカーネル2.4ではなく、カーネル2.2を継続して採用することもあり得る。これについては、カーネル2.4がいつごろ安定して使えるようになるか次第と考える」

レーザーファイブ

続くはLinuxディストリビュータの老舗であるレーザーファイブの代表取締役CEO・窪田敏之氏だ。Linuxが盛り上がる前から関わってきた同氏は、

「2001年は技術熟成の年」と語る。

「カーネル2.4の先進機能、XFree86 4.x、ThinkFreeOfficeにKOffice、Nautilus、Konqueror、それにDVDと、これ以上なにが必要なのというぐらい充実する。2001年は新しい技術を取り入れるよりも技術の成熟の年になりそうだ」

「当社はLinux Solution を提供するという理念のもと、①フラッグシップのLASER5 Linuxの開発②超高負荷・超高信頼性でインターネットサーバOSとしてすでに実績が蓄積しているFreeBSD③コンシューマ向け機能およびデスクトップ機能に特徴があるMandrakeのディストリビューションおよびパッケージ提供④マイクロサーバに象徴されるLinux組み込み事業⑤Linuxによるシステムソリューション、なかでも高速メールサーバを中心としたソリューションの提供などの事業展開を図る予定だ」

日本SCO

次は、商用UNIX「UnixWare」を持ち、同製品部門をCalderaが買収したことにより、一気に注目度が高まったSCOの日本法人である、日本SCO代表取締役サーバソフトウェア事業部長・麻生 誠氏のコメントだ。同氏には、今年から来年にかけての展望に関するコメントもいただいたが、ここではやはり注目度が高い、同社のUNIX戦略についてのコメントを抜粋してみたい。

「Calderaは第1四半期にはSCOのサーバソフトウェアとプロフェショナル事業を買収し、LinuxとUNIXのトップソフトウェア提供企業が誕生する。日本でも日本SCOを母体として日本法人を設立し、さらに充実した販売、サポート体制で活動する」

「Calderaの『Linux for Business』として評判の高いOpenLinuxとSCOで培ったUNIX技術を統合し、個人ユーザーが中心のクライアントOSからエンタープライズ分野でも安心して使えるハイエンドなサーバOSのLinuxを開発し、提供する。またRed Hat Linux、TurboLinuxなどすべてのLinuxを対象として管理できるLinux統合管理ソリューション『Volution』も販売を開始する。2001年はCalderaが日本で本格的に活動するので期待してほしい」

ミラクル・リナックス

ミラクル・リナックスは、ビジネスに使えるディストリビューションを提供するという明確な目標を掲げているわけだが、そのコメントもビジネス的観点に根ざした実践的なものである。

「今年に入り、メジャーなハードウェアベンダー、ソフトウェアベンダーがLinuxへの対応を強化したことで、Linuxは実運用の段階へ入った。来年は今まで不足していた技術者層の拡大と、Linux関連ビジネスの成長が期待できる。インターネット自体との親和性などからも、次世代のプラットフォームOSとして来年は、さらにシェアを伸ばし、PCサーバの20%はLinuxが使われると予想する。また、クラスタ対応などにより、高可用性、スケーラビリティのあるシステム構築など、よりハイエンドソリューションでのLinux導入も進むと見込んでいる」

「サーバ用途においては、ハードウェア企業からのフィードバックにより、ジャーナリングファイルシステムや、クラスタリング、その周辺機器のデバイスドライバが充実すると予想している。クライアント用途においても、徐々に個人デスクトップ用途の利用が始まるだろう。ただ、来年においては、個人の開発用途としての利用がまだまだ主流だろう」

「パートナー企業とのアライアンスをより一層強化していく。サポート体制の充実や教育ビジネスの立ち上げによるLinux技術者の拡充を狙う。また製品においてはスケーラビリティ／高可用性を実現するハイエンド向けの『Miracle Linux Enterprise Edition』を出荷、IA-64サーバ、カーネル2.4対応などを予定している」

オモイカネ

最後は、ビジネス用途に使いやすいディストリビューション「Omoikane GNU/Linux」を開発しているオモイカネ代表取締役・大熊但由氏のコメントだ。同社は、Distribution Builderという新製品をリリースしたという。

「来年は多くのIT関連ベンダーがより本格的にオープンソースに取り組む1年になると考えてる。オモイカネは、効率的にオープンソースの利用を促進するために、Distribution Builderという製品をリリースした。この製品を使えばきわめて安価に、どのベンダーでもオリジナルのディストリビューションを持つことが可能になる」

●

ほかのコメントもあわせて読むと、来年以降のキーワードがいくつも浮かび上がってくる。筆頭はもちろんカーネル2.4なのだが、そのほかにも組み込み、Office、ERPなどのビジネスソリューションと、幅が広い。1999年に認知され、2000年では当たり前の存在になったLinuxだが、2001年はサーバ分野は当然のことながら、それ以外でもさまざまに活躍するという予想が大勢を占めたといえる。

正しきタコへの道

初級Linuxer養成講座

第17回　シェルの小技(3)〜さらにコマンドラインの編集

前回は、コマンドラインでさまざまな「編集機能」を使えることを説明した。とくに、「ヒストリ」機能と組み合わせることで、以前に入力したコマンドラインを「再利用」できるので、入力の手間を大幅に省くことができる。前回までに説明した機能を知っていれば、十分に効率よくコマンドラインを使うことができるが、今回はさらに高度な使い方を説明することにしよう。

文：竹田善太郎
Text : Zentaro Takeda

前回のノートPCのハードディスク故障に引き続いて、今月に入ってからは、Linuxの実験用に使っているデスクトップマシンの調子がおかしくなった。ハードディスクやCPUなどの機能部分がおかしくなったのではなく、電源ユニットの排気ファンが**おかしな音**をたてるようになったのだ。

これは常識かもしれないが、CPUクーラーや電源ユニット、ケースなどに付属している**冷却ファン**は**消耗品**である。製品のメーカーやクラスによって異なるが、だいたい数万時間から十数万時間で**おしゃか**になってしまうのである。これは、半導体などの電子部品の寿命に比べて、はるかに短い。とくに、直径が5cm以下の小型のファンは、回転数が高いせいか、よく壊れるような気がする。いままでに経験したファンの故障は、すべて直径4cmのタイプで、外付けハードディクの冷却ファンに使われているものが多い。

今回おかしくなったPCは、いまではあまり見かけなくなった**NLX**タイプのもので、電源の冷却ファンも4cmのものが使われていた。これも常識かもしれないが、デスクトップPCはケースが大きいほどうるさいのではなく、逆に**小さいほどうるさい**ことが多い。使われている冷却ファンの大きさが小さくなるのに、内部で使われているCPUなどの発熱量は同じなので、より高い回転数でファンを回さなければ、冷却が間に合わなくなるからだ。だから、ファンの寿命も短くなってしまうわけだが、当然、メーカーはそのあたりを考えて、ファンの交換を簡単にできるように設計しているものだと思っていた。

ところが、いざケースを開いてみたところ、電源ユニットをばらさないと、ファンは交換できないようになっていた。この電源ユニットも、ねじを外しただけでは開かなくて、ケースをこじって**無理やり**開かなければならない構造になっている。ファンだけを簡単に交換できるようには設計されていないのだ（**写真1**）。ならば、電源ユニットごと交換できないかと思ったのだが、**NLX用の電源など、どこにも売っていない**のである。ATX用の電源ユニットなら、家電量販店のパソコン売り場でも手に入るようになったのに、やはりマイナーなNLXだけある。

CPUやマザーボードは健在なのに、数百円の冷却ファンひとつが壊れただ



写真1　NLX電源の冷却ファン
NLXタイプの電源には、何種類かあるようだが、これは細長い形をしたもの。危険防止の意味もあるのだろうが、ケースは簡単には開かないようになっていて、冷却ファンの交換は「事実上」不可能である。ファンが壊れたら電源ユニットごと交換するしかないのだが、肝心の電源ユニットが手に入らないので「仕方なく」ファンをむりやり交換することにした。感電や火災などのリスクが発生するので、このような「修理」はお勧めできない。

けで、通常の手段では修理不可能になるとは、ちょっと困ったことである。結局は、自分で組み立てたマシンなのでメーカー保証もなにもないわけだから、電源ユニットのケースをムリやりこじ開けて、新しいファンと交換した。しかし、このファンも安物しか手に入らなかったので、遅かれ早かれ、また壊れることになるだろう。NLXタイプのPCが絶滅に瀕している理由は、こんなところにもあるのだと思う。

カット&ペースト

前回は、主に「ヒストリ機能」について説明したあとで、「bashにはEmacsモードとviモードがある」ということに触れたところで終わりにした。この「Emacsモードとviモード」についての説明は最後にすることにして、bashのコマンドラインで使える編集機能について、もう少し補足しておこう。

コンピュータソフトに関する話をしている際にテキスト編集機能という言葉を聞いて、どんなことを思い出すかといえば、挿入、削除、検索などに加えて、カット&ペーストが挙げられるだろう。テキストデータの世界にとどまらず、グラフィックソフトなどでもカット&ペーストのテクニックを身につけるのは、いかに効率よくソフトを使いこなせるようになるかの重要な鍵である。bashのコマンドライン編集機能にも、簡単なものではあるがカット&ペースト機能が含まれている。

bashのオンラインマニュアルや参考書を見ても、「カット」や「ペースト」という言葉は見つからないかもしれない。その代わり、kill（キル）とyank（ヤンク）という単語で探してみると、いくつかの機能が見つかるだろう（**画面1**）。「kill」は「削除」と同じ意味だと想像がつくだろうが、「yank」とは聞きなれない単語で、英和辞典などで調べてみても、「仕事を引き上げる」などの意味しか載っていないと思う。これは、ことUNIXの世界ではよく使われている「業界用語」のようなもので、一般のコンピュータ用語でいうところの「ペースト」とか「貼り付け」と同じような意味である。この際だから、頭の片隅に置いておいてもよいだろう。

bashのコマンドラインでのカット&ペーストは、一般的なアプリケーションのカット&ペーストと同じように、削除（kill）したデータを一時的に蓄えるバッファ（bashの場合、kill-ring：キルリングと呼ばれている）に保存し、貼り付け（yank）操作を実行したときに、そのデータを取り出すようになっている（**図1**）。削除したデータがキルリングに保存されるのは、行単位や単語単位での削除を行う操作を行ったときで、1文字単位での削除（Ctrl-DやDeleteキーなど）を行ったときなどは保存されない。参考までに、カット&ペースト（キルアンドヤンク）操作の一部として使える削除関連コマンドを**表1**にまとめておく。

キルリングに蓄えられているデータをコマンドライン中に貼り付けるときは、Ctrl-FやCtrl-B（あるいは左右矢印のカーソルキー）を使って貼り付けたい位置にカーソルを動かしてから、Ctrl-Yを押せばよい（**図1**）。ちなみに「Y」は「Yank」の頭文字である。

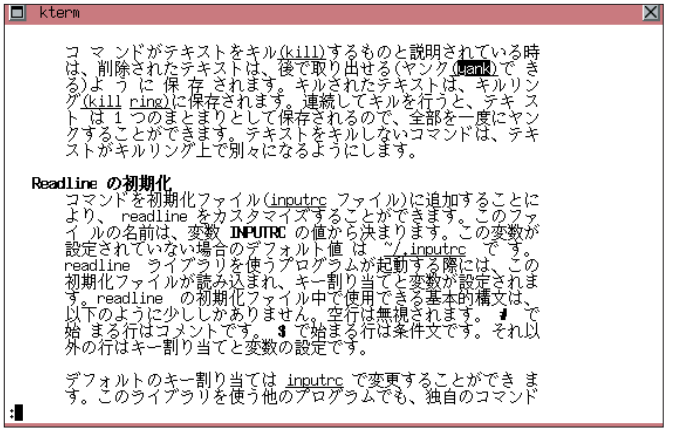
■キルリングを「回す」

ところで、一時保存用のバッファのことを「キルリング」と呼んでいるのには理由がある。bashのキルリングは、過去に削除したデータを10回分だけ保存してあって、それをあたかもリング（輪）を回すようにして呼び出すことができるのだ。ワープロソフトなどの「複数回アンドゥ」に似ていなくもないが、仕組みは若干違う。

キルリングの内容を呼び出す操作は、先に述べたようにCtrl-Yだが、いったん呼び出したあとで、ESC Yキーを押してみると、コマンドライン上に呼び出された部分が、次々と変わる。そして、ESC Yキーを10回以上押すと、以前に呼び出した文字列に戻る。ちょうど**図2**のように、リング状になったプレートを、ぐるぐる回している様子を思い浮かべればわかりやすいだろう。

■簡易アンドゥ機能として使う

このように、bashのカット&ペースト機能は、テキストエディタなどとほ



画面1　bashのオンラインマニュアルでカット&ペーストの説明を探す
bashのオンラインマニュアル（man bash）は、プリントアウトすると80ページを超えるほどの分量がある。目的の機能についての機能を見つけ出すのは、lessコマンドの検索機能を使ってもひと苦労だ。いわゆる「カット&ペースト」に関連する機能の説明を読みたい場合は、「yank」か「kill」という単語で検索してみるとよい。

ぼ同等の機能をもっているのだが、実際にはあまり頻繁に使うことはないかもしれない。コマンドライン上で、コマンド名やファイル名などをカット&ペーストすることは少なく、**ヒストリ機能や自動補完機能を使ったほうが早い**場合が多いからだ。しかし、これを**簡易アンドゥ**機能として使うと、意外に便利である。たとえば、あるコマンドラインを入力してから、**間違えたかも**……と思ってCtrl-Uキーなどでコマンド行全体を削除した後に、**やっぱり正しかった**と思い直したときなどは、コマンド行は実行する前にすでに削除してしまってあるので、ヒストリ機能などで呼び出すことはできない。このようなときには、Ctrl-Yキーを押してみれば、削除してしまったコマンドラインを一発で呼び戻すことができるのだ。

実は、後述するようにbashのコマンドラインには正式なアンドゥ機能もあるのだが、キー操作が面倒なので、筆者はこちらの「簡易アンドゥ」しか使っていない。

キー操作	動作
ESC Delete	カーソルより前の1語を削除
ESC Ctrl-H	〃
ESC D	カーソルより後ろの1語を削除
Ctrl-K	カーソル位置から行末までをすべて削除
Ctrl-U	カーソル位置から行頭までをすべて削除

表1　文字列データをキルリングに保存する削除コマンド（主なもの）

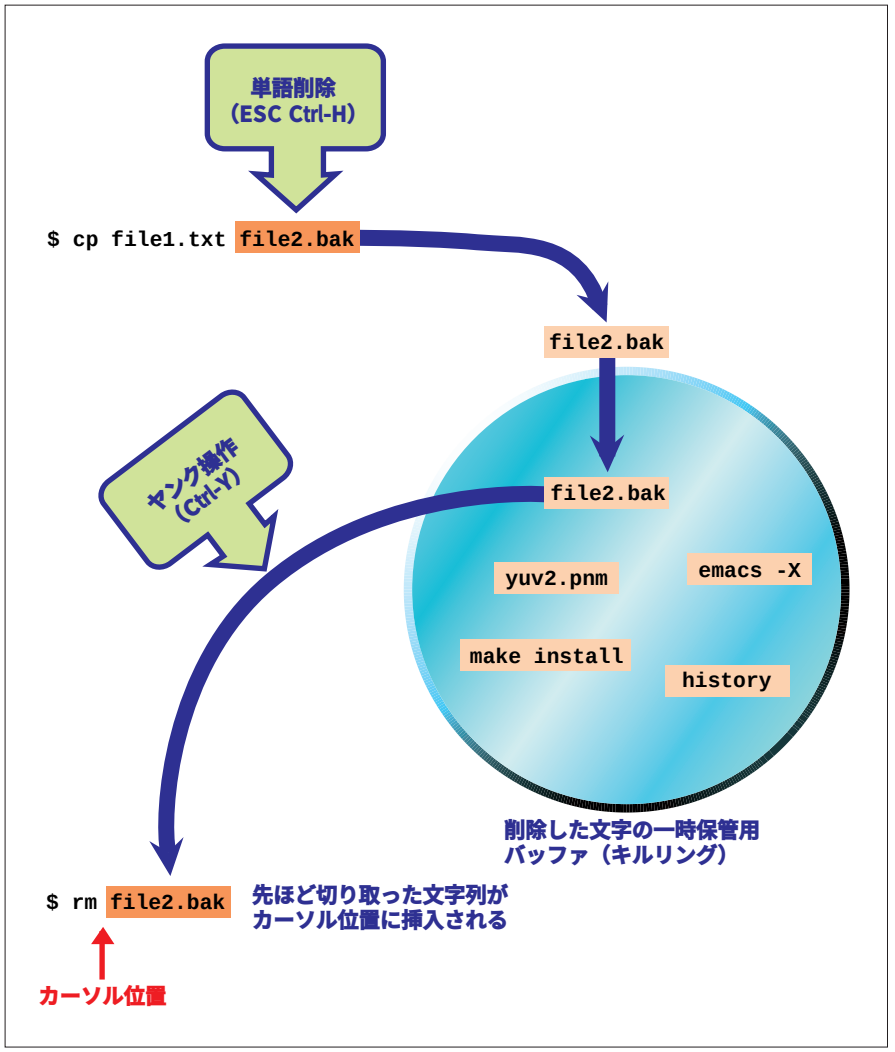


図1　bashのコマンドラインのカット＆ペースト機能の概要
単語削除や行削除などで削除された文字列データは、「キルリング」と呼ばれる一時保管場所に保存される。この状態で、Ctrl-Yキーを押すと、直前に行った削除操作で保存されたデータが、カーソル位置に挿入される。

アンドゥ機能

bashのコマンドラインの（正式な）アンドゥ機能は、Ctrl-_（コントロール＋アンダーバー）または、Ctrl-X Ctrl-Uで呼び出すようになっている。コマンドラインの編集中に、直前の操作（文字入力や削除など）を取り消したい場合には、これらのキー操作をすれば、以前の状態に戻してくれる。アンドゥのキー操作を繰り返せば、その回数だけ以前の状態に戻ってゆくのは、一般のアプリケーションソフトのアンドゥ機能と同じである。

しかし、「Ctrl-_」というキーには、**いったいどうやって入力すればいいのか**悩まされるのではないだろうか。正解は、CtrlキーとShiftキーの両方を押しながら、英語キーボードなら「－」（マイナス）キー、日本語キーボードなら「¥」（円記号）キーを押すのだが、キーボードの種類や環境設定によっては、これらのキーではうまく入力できない場合がある。また、Ctrl-X Ctrl-Uの場合は、2つのキー操作を続けて行う必要があるので、そのぶん余分な手間がかかる。もちろん、bashは設定ファイルを自分で作成すれば、キー操作のカスタマイズもできるのだが、そこまでしてコマンドラインでアンドゥ機能を使いたいと思う人は少ないだろう。それよりは、まず自動

補完機能やヒストリ機能に慣れたほうがよい。

文字列の置換

「ワードプロセッサ並みの編集機能があるなら、文字列の置換もできるのでは」と思うのが当然だろう。実際、bashにも文字列の置換を行う機能がある。たとえば、直前に入力したコマンドラインが次のような場合を考えてみよう。

```
$ man bash
```

ここで、bashではなくtcshのオンラインマニュアルを入力したい場合は、あらためてコマンドラインを入力してもたいした手間ではないが、次のようにコマンドラインの文字列の一部を置換して、目的のコマンドを実行させることもできる。

```
$ ^bash^tcsh
```

このコマンドの意味は、「直前のコマンドの中の“bash”という文字列を“tcsh”に置き換えて実行せよ」ということになる。同じ意味のコマンドを、次のように表わすこともできる。

```
$ !!:s/bash/tcsh/
```

UNIXやMS-DOSなどでsedコマンドを使ったことのある読者なら、**ああ、あれか**と心当たりがあるかもしれないが、これはsedコマンドの文字列置換表記と似ている。先頭の！！：の部分は、**直前のコマンドを対象に**という意味だと思っておけばよい。ちなみに、「！数字：」（たとえば、！5：など）とすると、**ヒストリ番号5番のコマンド**が対象になり、！－3：のように、負の整数を指定すると、**3回前に実行したコマンド**という意味になる。

「！！：s/文字列1/文字列2/」や「^文字列1^文字列2」という文字列置換表記では、**コマンド文字列中で、最初に登場する文字列1を文字列2に置換する**という意味になる（**図3**）。だから、たとえば、

```
$ cp file1.txt file1.bak
```

というコマンドの文字列を置換して、

```
$ cp file2.txt file2.bak
```

のようにしたい場合、

```
$ ^file1^file2
```

や、

```
$ !!:s/file1/file2/
```

では、

```
$ cp file2.txt file1.bak
```

のように置換されてしまって、都合が悪いことになる。

このような場合には、**文字列全体に対して置換を行う**、という意味を追加する**修飾子**というものを追加すればよい（**図4**）。ただし、「^文字列1^文字列2」の形式の置換コマンドでは、修飾子をつけることはできないので、「！！：s/文字列1/文字列2」の形式の置換しか使えない。

文字列全体の置換を行う修飾子は、置換コマンドの前にgという文字を追加すればよい。「g」は「global」の意

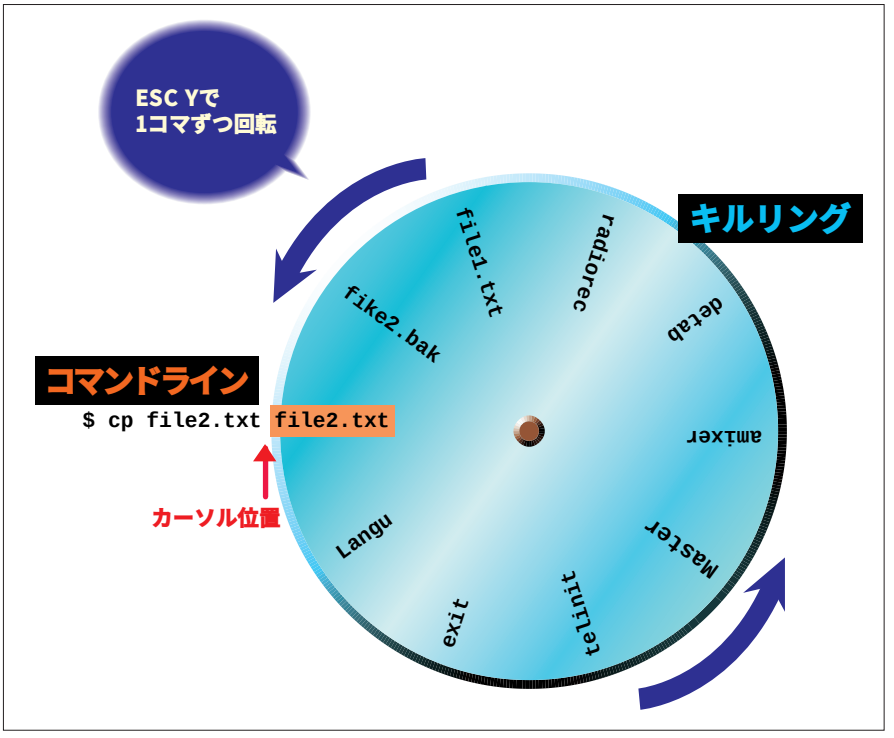


図2　キルリングの使い方
bashのキルリングには、最大10回分の削除データが、「リング状」になった保存場所（バッファ）に保管されている。Ctrl-Yを押すと、最新の削除データが呼び出されるが、ここでESC Yキーを押せば、あたかもリングを回転させるかのように、より古いデータが次々と呼び出されてくる。

味である。ところで、sedの場合は、「s/文字列1/文字列2/g」のように、最後に修飾子をつけるようになっているのだが、bashではそれとは異なるのに注意してほしい。

```
$ !!:gs/file1/file2/
```

こうすれば、目的のように、コマンド行の中のすべての「file1」という文字列が「file2」に置換されるようになる。

bashの文字列置換機能には、このほかにもさまざまな「修飾子」があったり、「何番目の文字列を置換するか」などを指定することもできるようになっているのだが、とりあえず以上で説明した方法だけ知っていれば、十分だと思う。くどいようだが、あまり細かい機能を覚えることよりも、自動補完やヒストリの使い方に早く慣れたほうがよい。

「コメント」を活用する

ヒストリの検索や置換をある程度使えるようになると、過去に実行した膨大な（標準の設定で1000行）コマンドラインの中から、目的のコマンドラインを見つけ出すのに苦労することもあるだろう。このようなときに備えて、覚えておくと便利なのが**コメント**の活用である。コマンドラインの先頭に#をつけて、自分の好きな文字列を入力してみよう。

```
$ # this is comment
```

普通なら、「command not found」などのエラーが出るはずだが、なにも起こらないはずだ。bashは、「#」以降の文字列を無視するようになっているからだ。そして、ここで入力した行は、すべてヒストリにも記録される。シェルスクリプトを書いたことのある読者ならご存知だろうが、このように「#」で始まる行のことを**コメント行**と呼んでいて、おもに、プログラムに「注釈」をつけて、読みやすくするために使われている。

「プログラムを書いているわけでもないのに、コメントなんてナンセンス」と思う読者もいるかもしれないが、ヒストリを検索しようとするときに、適当なコメントをはさんでおくと、意外と便利なことがある。たとえば、ある一連の作業を開始するときに、その作業について一言説明するコメントを入れておけば、あとでそのヒストリを検索するときに、キーワードとなる単語でから検索できるようになる。

```
$ # Linuxマガジン2月号の原稿用
$ tar czf genko.tgz /home/zen-t/lx200102
      :
      :
      :
$ # compiling ntpd
$ make depend
$ make
      :
      :
      :
```

このように入力しておけば、たとえば「Linuxマガジン」という文字列でヒストリを検索すれば、その仕事に関連して実行したコマンドのヒストリをすばやく検索できるようになる。該当するコメント行が見つかったら、あとは上下の矢印キーなどで、その前後に実行したコマンドラインをスクロール

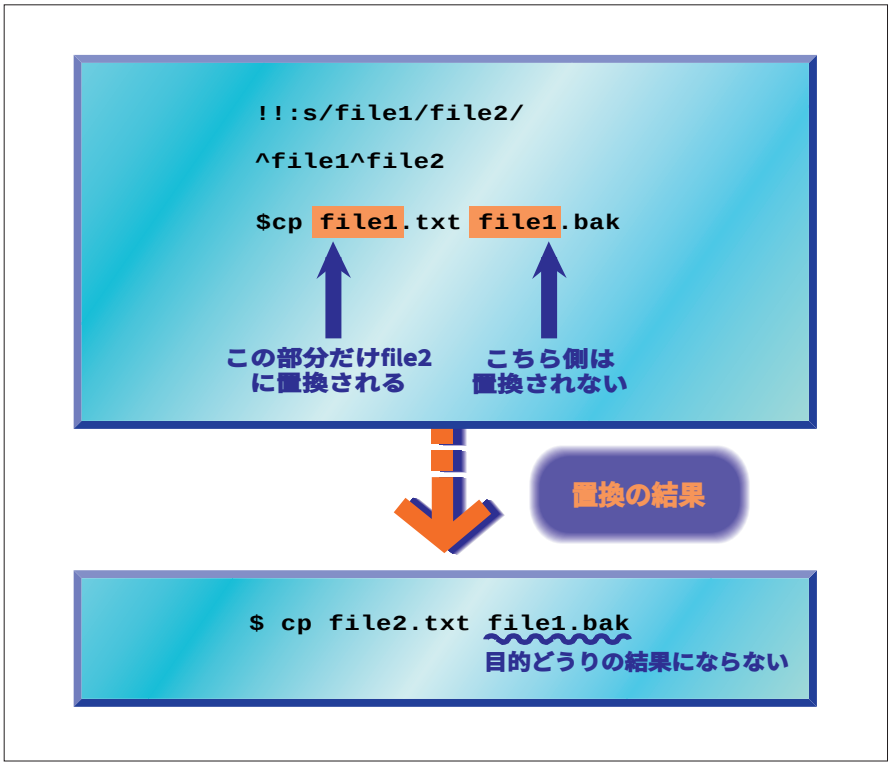


図3　「^文字列1^文字列2」あるいは「！！：s/文字列1/文字列2/」でコマンドラインの文字列置換を行った場合
この場合、文字列の置換は「1回限り」しか行われないので、該当する文字列が複数存在する場合も、最初の1つの文字列しか置換されない。

させて調べればよい。

ところで、コメント行を入力するときに、「#」の直後に「!」を入力してはいけない。「#!」という文字列は、「指定したコマンドにテキストをスクリプトとして渡す」という別の意味を持つからだ。

Emacsモードとviモードについて

いままで説明してきたヒストリ機能やコマンドラインの再編集機能の多くは、bashがEmacs編集モードになっているときに使えるものである。その名のとおり、Emacsで使われているカーソル移動や検索のキー操作がほとんどそのまま利用できるようになっているのだ。

しかし、bashでは、Emacsは嫌いとか大人ならviというユーザー向けに、vi編集モードというものも使えるようになっている。vi編集モードでは、テキストエディタのviと同じように入力モードとコマンドモードを切り替えながら、文字列の編集やカーソル移動を行うようになっている。もちろん、カーソルやヒストリの移動には、viと同じようにh、j、k、lなどのキーが使える。今回は誌面の都合で解説は割愛するが、bashをvi編集モードに切り替えるには、コマンドラインで次のコマンドを実行すればよい。

```
$ set -o vi
```

反対に、Emacs編集モードに切り替えるには、

```
$ set -o emacs
```

とする。

ところで、Emacs編集モードでコマンドラインを編集していると、Emacsに慣れているユーザーだと、つい編集結果を保存するくせでCtrl-X Ctrl-Sなどと押してしまうことがある。すると、その直後からキー入力がいっさいできなくなって、フリーズしてしまった！と思うことがあるかもしれない。しかし、これは単にターミナルエミュレータがCtrl-Sキーで一時停止状態に入っているだけなので、Ctrl-Qを押せばすぐに元の状態に復旧するので安心してほしい。

機能を覚えるのもほどほどに……

ここまでで説明してきた編集機能をすべて使えるようになれば、初級どころか上級のbashユーザーを名乗ってもおかしくない。しかし、bashには、1冊どころか数冊の本が書けるほど多くの機能が含まれていて、bashのオンラインマニュアルをプリントアウトしただけでも、A4判で80ページを超えるほどの大部になってしまうのだ。だから、Linuxのエキスパートであっても、bashのすべての機能をそらんじて使いこなせるような人間は、まず、いないと思う（もし本当にいたら、ごめんなさい）。

bashをはじめ、Linuxで使われている多くのコマンドは、世界中の無数のユーザーの要望を集めて作られ、改善されてきたものがほとんどだ。その結果として「機能のお化け」のようになってしまったコマンドも少なくないのだ。だから、膨大なページ数のオンラインマニュアルを前にして自分だけ使いこなせていないのではと悩む必要はまったくない。だれもが、自分が必要とする機能しか使っていないのである。

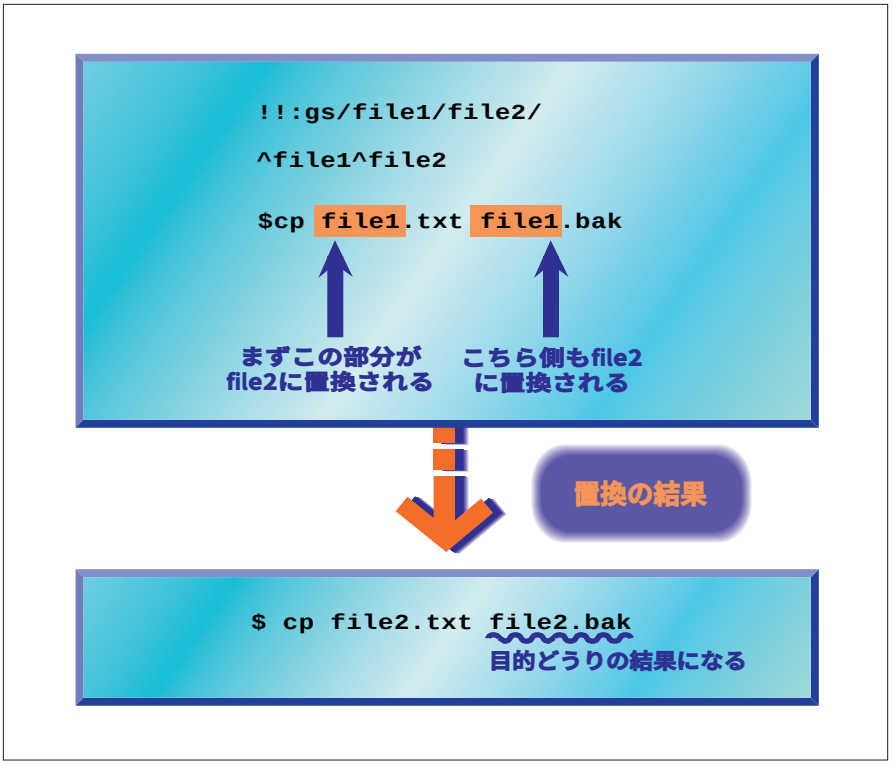
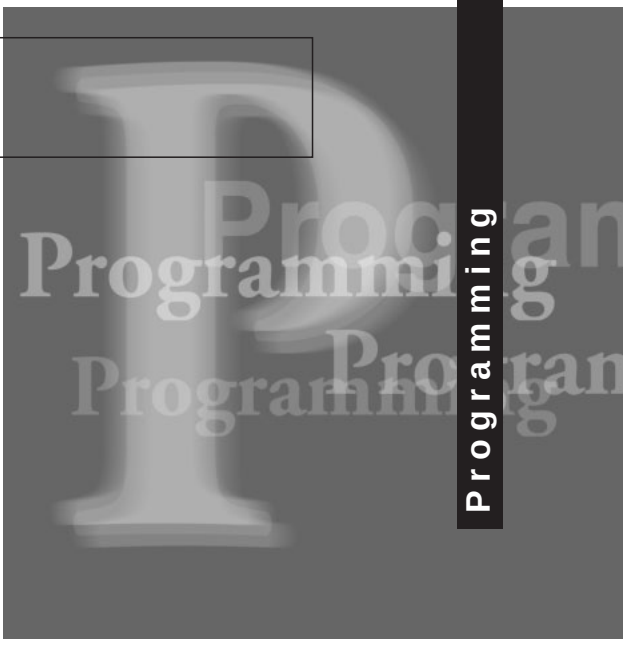


図4　「!!:gs/文字列1/文字列2/」で置換を行った場合
このように、すべての文字列を置換するように「g」という修飾子を追加すれば、該当するすべての文字列が置換されるようになる。ただし、「^文字列1^文字列2」という形式の置換では、このような方法は使えないことに注意。

プログラミング工房



先月号では、プログラムなどのバージョンを管理するシステムとして広く使われているCVSについての、最も基本的な部分についての具体例を説明した。今月は、ネットワークからの利用など、CVSの応用的な使い方について解説する。

第14回　バージョン管理の応用

文：藤沢敏喜
Text : Toshiki Fujisawa

多人数での開発

1人だけでプログラムを作っている場合は、自分の知らないうちにプログラムが勝手に書き換えられることはありえない。したがって、先月号で示したようにバージョン管理も比較的簡単である。

しかし、何人かで1つのプログラムを開発していると、自分の書いたソースコードを他人が変更するということが起きてくる。

小さな部屋で数名で開発している場合には、１人１人に声をかけても大した手間ではないので、「おーい。この部分を変更するぞー」と声を上げれば問題がない。けれど、30人くらいの人数での開発となると、ほかのスタッフが会議で席にいなかったり、休暇を取っていたりする場合があったりで、変更点を連絡することが容易ではなくなってしまうケースが多くなる。

また、最近の大規模なフリーソフトウェアの開発では、国境を越えて数百人規模で開発するケースも珍しくはない。このような場合、適切なバージョン管理をすることはきわめて難しくなる。

上記のようなインターネットを使った大規模な開発でも、CVSを使えば、最適なソリューションを提供することが可能になるのだ。

公開CVSサーバへのアクセス方法

先月号でも紹介したように、Apache、GNOME、FreeBSD、NetBSDなど無数のプロジェクトがCVSで管理され、広く一般に公開されている。

ここでは、ネットワーク経由でのCVSアクセスの実際例として、ソースコードのサイズが手頃なgPhotoのソースコードの取り出し方を説明してみよう。gPhotoは、デジタルカメラの画像データを取り扱うためのアプリケーションで、http://www.gphoto.org/に詳しい情報がある。

トップページからDOWNLOADをクリックすると、ダウンロード方法を記述したページに飛ぶことができる。そのページには、CVSによるプログラムソースコードの取得方法が書かれている。

gPhotoのソースコードのCVSは、GNOMEプロジェクトのマシン上で管理されていて、そのホスト名は、

```
anoncvs.gnome.org
```

である。anonは「anonymous」（匿名）の略であり、不特定多数の人からのアクセスを許しているCVSサーバとなっている。

また、CVSサーバへのアクセスには、/etc/servicesでcvspserverとして定義されているポート番号2401を使用

している。このため、CVSにアクセスするためには、このポート番号でアクセスできるようなインターネット接続環境が必要である。

しかし、企業や学校、そしてCATVによるインターネット接続サービスなどからのアクセスでは、ファイアウォールがあってこのポートが開いてないことも多い。そのような場合には、このCVSサーバへは直接アクセスすることができないので注意してほしい。

それでは、実際にgPhotoのプログラムソースコードを入手してみよう。

■環境変数CVSROOTの設定

先月号で説明したように、CVSの履歴はCVSリポジトリと呼ばれる場所に保存される。そして、この場所を指定するのがCVSROOTという名称の環境変数である。

先月号では、ローカルマシン（自分のマシン）のホームディレクトリにCVSリポジトリを設定したので、

```
$ export CVSROOT=/home/fujisawa/CVSROOT
```

というような設定を行ったが、gPhotoのプログラムソースは、anoncvs.gnome.orgというマシンの/cvs/gnomeというディレクトリに保存されているので、

```
export CVSROOT=\
":pserver:anonymous@anoncvs.gnome.org:/cvs/gnome"
```

というようにCVSROOTを設定する（誌面の都合で2行にしているが、「=」のあとに「\」なしで続けて1行で書いてもよい）。

ここで、「:pserver:」の部分は通信するプロトコルの指定であり、「anonymous」はCVSサーバにログインするためのユーザーIDである。anonymousを入力することにより匿名ログインが可能になり、パスワードなしでCVSにアクセスすることが可能になる。もちろんだれでも変更できては困るので、anonymousでログインした場合はコミット（書き込み）ができないようになっている。

■ログイン

CVSの内容を変更できる人のことを「コミッター」と呼ぶことがある。ローカルのハードディスクで自分1人で使っている場合には、コミッターは自分自身であるため、認証を行う必要はない。しかし、ネットワーク経由では、アクセスしてきた人が正規のコミッターであるか、そうでないかを認識する必要がある。このため、最初にCVSサーバにアクセスする場合には、ログインをしなければならない。ログインするためには、

```
$ cvs login
```

というコマンドを実行するとよい。このコマンドを実行すると、

```
(Logging in to anonymous@anoncvs.gnome.org)
CVS password:
```

というメッセージが表示され、パスワードの入力画面となる。ここでは、コミッターとしてではなく、匿名（anonymous）ユーザーとしてログインするので、ただ単にリターンキーを入力すると、ログインが終了する。

ログインが行われると、ホームディレクトリにある「.cvspass」というファイルにパスワードが保存されるので、2回目からはログイン作業を行う必要はなくなる。

ちなみに、

```
$ cvs logout
```

を実行すると、「.cvspass」からパスワード情報が消える。セキュリティのためには、まめにログアウトすることが望ましい。

■チェックアウト

ログインが正常に終われば、すべてのCVSコマンドが使えるようになる。したがって、先月説明したように、CVSリポジトリからgphotoのソースを取り出すためには、

```
$ cvs checkout gphoto
```

を実行すればよい。このコマンドの実行により最新バージョンのプログラムソースコードを取り出すことができる。なお、低速なネットワーク経由で取り出す場合は、通信時の圧縮レベルを指定するオプションを付けるとよい。圧縮レベルには1から9までの段階がある。「- z1」が低圧縮、「-z9」が高圧縮を意味するが、サーバへの負担を考えて、

```
$ cvs -z3 checkout gphoto
```

あたりを指定するとよいだろう。このコマンドを実行すると、次のような表示が行われ、カレントバージョンのソースコードを取り出すことができる。

```
cvs server: Updating gphoto
U gphoto/.cvsignore
U gphoto/ABOUT-NLS
U gphoto/AUTHORS
U gphoto/BUGS
U gphoto/CONTACTS
U gphoto/COPYING
U gphoto/DEPENDS
U gphoto/FAQ
U gphoto/HACKING
U gphoto/INSTALL
U gphoto/IOLIBRARY
U gphoto/LETTER
U gphoto/MANUAL
U gphoto/Makefile.am
U gphoto/NEWS
U gphoto/NEWS_0_4_0
U gphoto/NEWS_0_4_1
U gphoto/NEWS_0_4_2
U gphoto/NEWS_0_4_3
U gphoto/PROGRAMMERS
U gphoto/README
U gphoto/SUPPORTED
U gphoto/THANKS
     :
```

また、cvs logを実行すると、履歴情報が表示されるので、それを参照して特定のバージョンを取り出すこともちろん可能である。

たとえば、gphoto-0-3-4というタグのついたバージョンを取り出すためには、

```
$ rm -rf gphoto
$ cvs -z3 checkout -r gphoto-0-3-4 gphoto
```

とすればよい。なお、先月も述べたが「-r」を指定する場合は、「sticky tag」が貼り付かないように、「-p」オプションを指定するか、上記のように現在あるディレクトリを消してから取り出すことに注意が必要である。

ここでは、gPhotoの例について説明したが、ほかの公開CVSサーバでからも同様な方法で、最新バージョンのソースコードを取り出すことが可能である。しかしながら、必要もないのにむやみやたらに通信を行うことは、サーバに負荷がかかったり、ネットワークトラフィックの面から好ましくない。単にテストを行いたいときは、後述のように自分でCVSサーバを立てて実験してみてほしい。

CVSサーバのセットアップ

公開CVSサーバへのアクセス方法についてわかったところで、今度は自分でCVSサーバを立ち上げる方法について解説してみる。

1人だけで使う場合でも、CVSサーバを立てておくと何かと便利なこともある。しかし、CVSサーバが有効なのは、やはり複数名でプログラムを開発する場合である。

ここでは、佐藤さんと鈴木さんの2人がプログラムを開発する場合の具体例について述べてみたい。

■ユーザーの登録

まずは、CVSサーバに、CVSを使用するユーザーを作成する。ここでは、佐藤さん（sato）と鈴木さん（suzuki）を登録することにする。

■/etc/groupの設定

特定のユーザーだけが、CVSリポジトリを変更できるようにして、プロジェクトに関係のない人が勝手に変更できないようにするためには、グループパーミッションを適切に設定する必要がある。

このため、cvsという名称のグループを作成することにする。このグループにsatoとsuzukiを所属させる場合は、/etc/groupに、

```
cvs:*:1000: sato,suzuki
```

という記述を行えばよい。上記の1000はグループID番号で、/etc/groupでほかに使用していない番号を使う。

なお、匿名アクセスを許す場合は、上記のcvsグループにanonymousユーザーを追加する必要があるが、詳しくは先月紹介した参考文献などを読んでみてほしい。

■CVSリポジトリの作成

CVSサーバにするには、そのCVSサーバ上の適当なデ

ィレクトリにCVSリポジトリを作成する必要がある。たとえば、/var/cvs-repに作成する場合は、スーパーユーザーになってから、

```
# mkdir /var/cvs-rep
```

として、そのディレクトリを作成する。そして、

```
# cvs -d /var/cvs-rep init
```

を行い、リポジトリの初期化を行う。

次に、

```
# chown -R root:cvs /var/cvs-rep
```

を行い、$CVSROOT、つまり/var/cvs-repディレクトリの所有者を「root」にし、そのディレクトリのグループを先ほど定義した「cvs」とする。

なお、「-R」オプションは、指定したディレクトリ下にあるすべてのファイルを指定したユーザーとグループにするオプションである。ユーザーとグループを区別するために「.」を使うこともできるが、最近のUNIX（POSIX）では、ユーザー名にピリオドが使えるように「:」を使うようになってきている。

そして最後に、/var/cvs-repとその下のCVSROOTのパーミッションを、

```
# chmod 770 /var/cvs-rep
```

▼リスト1　CVS練習用の3ファイル

```
=== myprog/main.c ===
extern void disp(void);
int main(void)
{
        disp();
}

=== myprog/Makefile ====
myprog: main.c func/disp.c
        cc -o myprog main.c func/disp.c

=== myprog/disp/func.c ===
#include <stdio.h>

void disp(void)
{
        printf("Hello World\n");
}
```

```
# chmod 770 /var/cvs-rep/CVSROOT
```

として、cvsグループに所属するユーザーが読み書き可能になるように設定する。

■/etc/inetd.confの設定

先ほど説明したように、CVSサーバは/etc/servicesでcvspserverという名称で定義される2401番ポートを通してアクセスされる。したがって、このポートでアクセスされた場合に、cvsコマンドが適切に起動されるように設定をする必要がある。このためには、/etc/inetd.confに以下のような記述を追加してリブートすればよい（あるいは、kill -HUPを使って、inetdデーモンにハングアップシグナルを送る）。

```
cvspserver stream tcp nowait root /usr/bin/cvs cvs
--allow-root=/var/cvs-rep pserver
```

以上でCVSサーバのセットアップは終了である。あとは先月解説したのと同じ要領で、CVSの機能を使うことが可能になる。

プログラムソースの登録

ここでは実験のため、今立ち上げたCVSサーバに、先月と同じプログラムソース（**リスト1**）を、佐藤さんが登録する例を紹介してみる。まずは、CVSサーバとは別のマシンにユーザー名satoでログインしよう（ここでログインするユーザー名は必ずしもsatoでなくてもよく、CVSROOTで指定したユーザー名で認証される）。

次に、CVSROOTという環境変数を設定する。今回はネットワーク経由でのアクセスになるので、接続するプロトコルとサーバ名およびユーザー名を、次のように指定する（~/.bash_profileなどに書いておくとよい）。

```
export CVSROOT=\
":pserver:sato@cvs.xxxx.co.jp:/var/cvs-rep"
```

ここで「pserver」が認証プロトコル名で、「sato」がユーザー名、そして「cvs.xxxx.co.jp」がCVSサーバのホスト名である。また「/var/cvs-rep」がCVSリポジトリがあるディレクトリである。

さて、環境変数の設定が終わったら、次はCVSログイ

ンを行う。gPhotoのときと同じように、

```
$ cvs login
```

として、CVSサーバに登録してある佐藤さんのパスワードを入力すればOKである。

次は、プログラムソースファイルを登録してみよう。/home/sato/myprogに**リスト1**のファイルを用意し、先月号と同じように、

```
$ cvs import -m "comment" myprog v-tag r-tag
```

を実行すると、

```
N myprog/main.c
N myprog/Makefile
cvs server: Importing /var/cvs-rep/myprog/disp
N myprog/disp/func.c

No conflicts created by this import
```

というメッセージが出力され登録が行われるはずである。

プログラムソースの取り出し

ここまでで、CVSサーバへのプログラムソースコード登録を行うことができた。今度は、鈴木さんが自分のLinuxマシンから、CVSサーバに登録されたプログラムソースコートを取り出す例について説明しよう。

例によって、環境変数を次のように登録する。

```
export CVSROOT=\
":pserver:suzuki@cvs.xxxx.co.jp:/var/cvs-rep"
```

ここで、ユーザー名は上記のようにsuzukiを指定することに注意しよう。そして、

```
$ cvs login
```

として、CVSログインを行い、

```
$ cvs checkout myprog
```

とすれば、カレントディレクトリにプログラムソースコードを取り出すことができる。

なお、新しいファイルの追加や削除、そしてログの参照や、特定のバージョンの取り出しについては、先月説明したのとまったく同じ操作で行うことができる。

複数人でのファイル格納方法

複数のファイルを複数の人で開発する場合、1つのファ

Column

CVSサーバ用として適切なOS

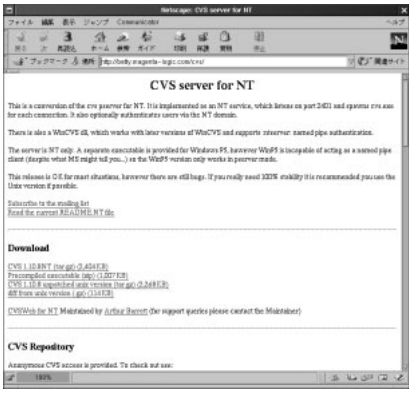
筆者の場合、大事なソースファイルを置くCVSサーバとしては、FreeBSDだけしか使用していない。しかし、Windows NT／2000用で動作するCVSサーバというのも存在するようである。

Windows NT／2000用CVSサーバを提供しているページである、http://betty.magenta-logic.com/cvs/（**画面1**）によると、

This release is OK for most situations, however there are still bugs. If you really need 100% stability it is recommended you use the Unix version if possible.

という文章が続き、NT版にはバグがあるので、100%安定した動作を望む場合は、UNIXを使うことが推奨されている。筆者自身も、Windows 2000上で重要なソースファイルを管理するのは非常に抵抗を感じる。どうしてもWindows 2000のマシンを使わなければならない状況に追い込まれた場合は、このCVSサーバを使うより、Windows 2000でVMwareを動作させ、その上でLinuxを動かしたほうが、ずっとましなのではないかと思ってしまう。

というわけで、CVSサーバを動作させるのに適切なOSは、やはりFreeBSDやLinuxなどの安定したUNIX系と言わざるをえないようである。



画面1　CVS server for NTのWebサイト

イルを編集する人が特定の1人だけであれば、コミットする場合に問題が起きることはない。

しかしながら、2人が同時に同じファイルを編集し、コミットしようとすると問題が起きることがある。

たとえば、main.cを佐藤さんが、

```
extern void disp(void); /* add by sato */
int main(void)
{
        disp();
}
```

のように変更してコミットしたあとに、鈴木さんが、

```
extern void disp(void);
int main(void)
{
        disp(); /* add by suzuki */
}
```

のように変更してコミットしようとすると、問題が起きる。

この場合、鈴木さんが、

```
$ cvs commit -m "comment by suzuki"
```

を行うと、

```
cvs commit: Examining .
cvs commit: Examining disp
cvs server: Up-to-date check failed for `main.c'
cvs [server aborted]: correct above errors first!
```

となって、鈴木さんのホームディレクトリにある「main.c」が、CVSサーバのリポジトリにある最新のソース、つまり佐藤さんが更新した「main.c」と整合しないことが報告される。

佐藤さんが、このエラーを解消するためには、

▼リスト2　2人の追加が反映されたmain.c

```
extern void disp(void); /* add by sato */
int main(void)
{
        disp(); /* add by suzuki */
}
```

```
$ cvs update
```

を実行する必要があるが、このコマンドを実行すると、次のような表示が出力される。

```
cvs server: Updating .
RCS file: /var/cvs-rep/myprog/main.c,v
retrieving revision 1.1.1.1
retrieving revision 1.2
Merging differences between 1.1.1.1 and 1.2 into
main.c
M main.c
cvs server: Updating disp
```

ここで、main.c の左に表示される「M」はMergeを意味し、佐藤さんの変更と鈴木さんの変更がマージされたことを示している。

この結果、鈴木さんのホームディレクトリにあるmain.cは、**リスト2**のように佐藤さんと鈴木さんがそれぞれ追加したコメントの両方が適切に反映される。

ここで、鈴木さんが再び、

```
$ cvs commit -m "comment"
```

を行うと、今度はエラーが起きずに、次のように表示され、正常にコミットが実行される。

```
cvs commit: Examining .
cvs commit: Examining disp
Checking in main.c;
/var/cvs-rep/myprog/main.c,v  <--  main.c
new revision: 1.3; previous revision: 1.2
done
```

ここまでの段階で、リポジトリは**リスト2**の状態に変更されることになる。

ところで、この時点で佐藤さんのホームディレクトリは、**リスト2**の古い状態のままになっている。佐藤さんが、鈴木さんのこの変更を自分のホームディレクトリに取り込むためには、同じように、

```
$ cvs update
```

を実行すればよい。すると、次のように表示される。

```
cvs server: Updating .
P main.c
cvs server: Updating disp
```

ここで、main.cの左の「P」は、自分（佐藤さん）のディレクトリにあったファイルよりも新しいリビジョンのファイルがリポジトリにあり、両者の差分情報（Patch）が送られ、自分のディレクトリにあるmain.cが新しいリビジョンに更新されたことを意味する。ここで、「P」はPatchの「P」を意味する。

なお、「P」とほぼ同じことを意味する「U」が表示されることもあるが、これはファイルの差分が大きすぎる場合で、パッチではなくファイル全体が送信される。「U」は、Updateの「U」である。

コンフリクト（衝突）

前の例では、佐藤さんと鈴木さんが変更した部分が違う行であったため、CVSが適切に判断しマージを行うことができた。

しかし、同じ行を2人が同時に編集した場合はどうなるだろうか？ たとえば、佐藤さんが、

```
int main(void) /* by sato */
```

と変更し、鈴木さんが、

```
int main(void) /* by suzuki */
```

と変更したようなときである。

この場合、まず佐藤さんがコミットを行うと、リポジトリは、

```
extern void disp(void); /* add by sato */
int main(void) /* by sato */
{
        disp(); /* add by suzuki */
}
```

という状態になる。

この段階で鈴木さんがcvs updateを行うと、次のようなエラーが報告される。

Column

Windows上での開発におけるCVSの利用

組み込み用機器で使うCPUなどでは、Windows用のコンパイラしか用意されていないため、Windowsを使わざるをえない場合も少なくない。しかしながら、バージョン管理だけは安定したUNIX上でCVSを使いたいというのが開発者としての偽らざる気持ちである。ここで、CVSサーバとしてUNIXを使う場合、クライアント側では次のような開発方法が考えられる。

1. WinCVS（またはcygnus環境）を使用する
2. WindowsのファイルをsmbmountなどでUNIXからマウントする
3. UNIX上のファイルを、sambaでWindowsからマウントする
4. Wine（Windowsエミュレータ）を使い、すべてUNIX上で開発する

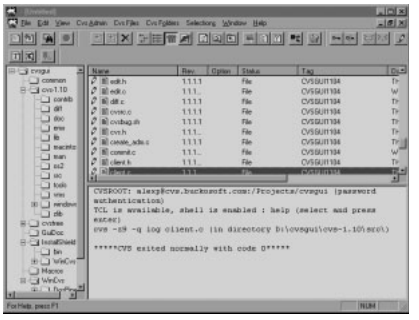
ここで、1は、Windows上のGUIプログラムとして作成されているWinCVS（**画面2**）を使用し、Windowsのドライブ上で、直接チェックアウトしたりコミットしたりする方法である。WinCVSは昔ほんの少しだけ評価したことがあるが、筆者は服の上から背中を掻いているようなもどかしさを感じた。

2の方法は、高速にコンパイルが可能であるものの、Windows上にファイルを展開するというのが筆者には気持悪く感じられ、実際に運用したことはない。

3は実際に数十人規模の開発に使ったことがある方法だ。UNIX上でコマンドを使わなければならないので、Windowsしか使ったことのない人が多いと敷居の高い方法ではあるが、適切に教育を行ったため特に問題はなかった。

4はまだ実際の開発で使ったことはないが、かなり期待が持てる方法ではないかと思う。最近のWineは、コンパイラのようなコマンドラインアプリケーションを実行するには十分実用的で、組み込み用CPU向けの某コンパイラをWineで実際に動作させたところ、問題なく動作し、速度も満足できるものであった。

UNIX使いにとって、Windows上での開発はできれば避けたいものである。ソニーのPlayStation開発環境のように、Linuxを使った開発環境が1日でも早く普及することを願いたい。



画面2　WinCVSの実行画面

```
cvs server: Updating .
RCS file: /var/cvs-rep/myprog/main.c,v
retrieving revision 1.3
retrieving revision 1.4
Merging differences between 1.3 and 1.4 into main.c
rcsmerge: warning: conflicts during merge
cvs server: conflicts found in main.c
C main.c
cvs server: Updating disp
```

上記では、main.cでconflictが見つかったことが示され、main.cの左には衝突（Conflict）を示す「C」が表示されている。

このupdateで、鈴木さんのディレクトリにあるmain.cは、**リスト3**のようにCVSが自動的に書き換えてしまう。

リスト3で、

```
<<<<<<< main.c
int main(void) /* by suzuki */
```

の部分が、自分のディレクトリにあったmain.cの内容であり、

```
=======
int main(void) /* by sato */
>>>>>>> 1.4
```

の部分が、リポジトリにあるリビジョン1.4の状態であることを示している。

この場合、先ほどのように、CVSが自動的に判断することはできないため、鈴木さんはmain.cを手動で書き換える必要がある。つまり、

```
<<<<<<< main.c
int main(void) /* by suzuki */
=======
int main(void) /* by sato */
>>>>>>> 1.4
```

という部分をエディタで、

```
int main(void) /* by sato and suzuki */
```

と書き換える。そしてコミットを行うと、

```
cvs commit: Examining .
cvs commit: Examining disp
Checking in main.c;
/var/cvs-rep/myprog/main.c,v  <--  main.c
new revision: 1.5; previous revision: 1.4
done
```

というように、リビジョン1.5として正常に格納されることになる。

▼リスト3　CVSによって書き換えられた鈴木さんのmain.c

```
extern void disp(void); /* add by sato */
<<<<<<< main.c
int main(void) /* by suzuki */
=======
int main(void) /* by sato */
>>>>>>> 1.4
{
        disp(); /* add by suzuki */
}
```

おわりに

多くのプロジェクトのプログラムソースコードが、CVSによって公開されている。今月説明したように、公開CVSからのファイルの取り出し方は難しくない。これを覚えれば、いつでも最新ソースコードが入手できてたいへん便利である。

大規模に多人数で開発を行うことは、たいへん難しい作業であるが、CVSを使うことによって効率的な開発を行うことができる。複数の人でCVSを使うのはちょっと面倒にみえるかもしれないが、今月号の記事を見ながら実際に自分で試してみると理解が深まるだろう。

なお、数十人規模でCVSを使った開発をするとさまざまなトラブルが発生しがちである。複数人での本格的な運用を始める前に、半年くらいは一人だけでCVSを実際に使ってみるのが安心だろう。

また、たくさんの人が使うと慣れないうちはリポジトリをグチャグチャにしてしまう場合もあり、リポジトリを前の日の状態に戻したいことがある。したがって、毎朝テープに自動的にバックアップするように設定しておくことは非常に重要である。

Rubyで行こう

2000年11月29日～12月1日に京都で「Perl/Ruby Conference」（「Linux Conference 2000 Fall」併催）が開催されました。Perlの父Larry Wallさんと、Rubyの父まつもとさんの両巨頭が対面したカンファレンスの模様を、Ruby中心の視点でレポートします。

第14回　ツインピークス

文：赤松智也
Text : Tomoya Akamatsu

11月29日から12月1日までの期間、京都国際会議場において「Linux Conference 2000 Fall」と、それに併催された「Perl/Ruby Conference」が開かれました（**写真1**）。今回のカンファレンスには、Larry Wallさんとまつもとさんの2人が出席（**写真2**）しただけでなく、お互いが直接対話するセッションも用意されました。今回は彼らのほかにも、『伽藍とバザール』で知られるEric Raymondさんなど、たくさんの有名人が参加していました。

会場では、展示会と無料セッションや、有料のチュートリアルが行われました。展示会ではLinux関係企業によるブース展示や、Perl/Ruby Conferenceの主催であるオライリー・ジャパンによる展示即売会も行われていました。オライリーのブースでは、Larryさんやまつもとさんによるサイン会も行われました（**写真3**）。筆者もさっそくサイン本をゲットしました。これは読者へのプレゼントにしたいと思います。

無料のセッションも充実していましたが、有料のチュートリアルも『Rubyプログラミング入門』著者の原信一郎さんによる「入門Ruby」や、『プログラミングPerl』訳者の近藤嘉雪さんによる「入門Perl」など盛りだくさんでした。経済的事情から（笑）、有料のチュートリアルには参加できませんでしたが、参加したかったなあ（**表1**）。

今回のPerl/Ruby Conferenceでは、聞きたいセッションが重なることが多かったのが残念でした。同時に4つも5つものセッションが開かれていたので、当然と言えば当然なのですが、今度開かれるようなことがあれば、もう少し改善を希望したいです。



写真1　地下鉄烏丸線国際会議場駅改札を出てすぐ。看板がカンファレンスムードを盛り上げます



写真2　キモノガールをはさんで並ぶLarryさんとまつもとさん。2人ともうれしそう？

対談：PerlとRubyの比較

初日午前中に会場に着くと、さっそくまつもとさんとLarryさんによる直接対決の会場に移動しました。本当は朝9時から「入門Ruby」をやってるのですが……。

対決（？）は、まつもとさんが、Larryさんの持っていたRubyの本（Programming Ruby）にサインするという友好的な雰囲気で始まりました。意外というか当然というか、ほとんど対立らしい対立もなく、なごやかに進みました（**写真4**）。

まず、以前から「Larry Wallは私のヒーローである」と公言するまつもとさんによって、彼のすばらしい点が紹介されます。曰く、

基調講演、スペシャルセッション

■基調講演：Larry Wall氏（Perl作者）
　　まつもとゆきひろ氏（Ruby作者）
　　Eric Raymond氏（『伽藍とバザール』著者）
■SS：「各スクリプト言語の比較」
■SS：「ワンライナーのすすめ＋Perl/Rubyの鉄人」
　ワンライナーのすすめ：Tokyo Perl Mongers 前田薫
　Perlの鉄人、Rubyの鉄人
　鉄人の実況・解説：前田薫、まつもとゆきひろ
■SS：「オープンソースとビジネス パネルディスカッション」
　チェア：日本Linux協会 理事 佐渡秀治
　パネラー：Project BLUE、株式会社グッデイ 代表取締役 前田青也
　　三菱電機株式会社 藤巻淳
　　オムロンソフトウェア株式会社 中西紀晶
　　まつもとゆきひろ

==========
Perl/Ruby Conference チュートリアル一覧

■「入門 Ruby」原信一郎
■「入門 Perl」有限会社コンテキスト 代表取締役 近藤嘉雪
■「Advanced Perl」Tokyo Perl Mongers 台場圭一
■「Advanced Ruby：RubyUnit（Testing Framework）」Just another Rubyist 助田雅紀、Pair Programmer 石井勝
■「正規表現」Tokyo Perl Mongers 前田薫

==========
Perl/Ruby Conference カンファレンス一覧

■「CPANモジュールの使い方」川合孝典
■「Perl日本語情報処理」株式会社オン・ザ・エッヂ 最高技術責任者 小飼弾
■「WWWサーバサイドプログラミング」ゼータビッツ株式会社 ネットワーク研究所平研究員 前田修吾
■「ケーススタディ：Rubyで数値処理 ― NumRubyへ向けて」北海道大学 大学院理学研究科研究生数学専攻 後藤謙太郎
■「Porting Ruby ― Rubyの移植」Just another Ruby porter わたなべひろふみ
■「RubyとXML」INSエンジニアリング株式会社 事業開拓室 吉田正人
■「dRubyによる分散オブジェクト環境」Just Another Ruby Hacker 関将俊
■「Perl 5.6の新機能」有限会社コンテキスト代表取締役 近藤嘉雪
■「Perlによるwebアプリケーション開発の傾向と対策」株式会社クレイフィッシュ 運用部社員 小山浩之
■「モジュールの作り方」株式会社オン・ザ・エッヂ 最高技術責任者 小飼弾
■「Ruby開発者と話そう」Ruby設計者、開発者 まつもとゆきひろ
■「Perl & XML」Tokyo Perl Mongers所属 台場圭一

==========
Perl/Ruby Conference ワークショップ、BOF

■「Rubyにおける新しいGCの実装」広島市立大学 木山真人

表1　Perl/Ruby Conferenceプログラム

ひとつのイノベーションを実現するだけでもたいへんなのに、Larryさんは少なくとも3つものイノベーションを実現している。それは、patchとmetaconfig、そしてPerlだ。

たしかに、patchは今でも多くの人が日々使っていますし、metaconfigの発想を受け継いだautoconfもほとんどのフリーソフトウェアに用いられています。また、Perlがスクリプト言語という分野に新境地をもたらしたことに、同意しない人は少ないでしょう。Larryさんの書いたプログラムの多くが持つ人間味あふれるインターフェイス（彼はhuman interfaceと呼んでいました）についても紹介されました（たとえば、以前のpatchは「うーん、これはコンテキストdiffみたいだね。n行ずれてるけど当てておくよ」などと出力していました）。

しかし、本来は対決が行われそうなシチュエーションで、これだけ相手をほめちぎるとは、よっぽどあこがれているんでしょうねえ。

あとは友好的かつ熱い対話が行われました。以下はその抜粋です。

まつもと：RubyはPerlからたくさんの思想を受け継いでいます。
Larry：今朝ほんの少し本を読んだ限りでは、2つの言語はプログラマーに自由を提供するという点において似ていますね。
まつもと：RubyはSmalltalkのあたりから出発して、Perlに向かって成長してきました。
Larry：Perl5からオブジェクト指向機能があるが、クラスを使うことはともかく、クラスを作ることは「可能なこと」であって、「簡単なこと」ではありませんでした。Perl 6では、「簡単なこと」になるかもしれません。
Larry：Perl 6は完全な再書き換えになります。Perl 6では、内部的にはすべてのデータはオブジェクトになります。実装をC++で行うか、Cで行うかはまだ決めていません。
Larry：文字列と数値が相互に自動変換されるのは、AWKに汚染されすぎていたかもしれません。ほかにも私の愚かさが原因の問題はいくつかあります（Larryさんって、とても謙虚な人ですね）。
まつもと：それは私も同じです。Rubyの最大の長所は、Perlより若いことでしょう。Perlの問題点を見て、それを避けることができたから。
Larry：Perl 6はコミュニティベースの開発体制に移行しますが、言語設計者としての私の位置は残るでしょう。
まつもと：もう自分でぜんぶやらないんですか？
Larry：Perlはもう私1人の手に負えるサイズじゃないし、歳をとるとだんだん衰えるからねえ。
まつもと：（さみしそう）
Larry：Perl 6はメタ言語としての性質も持ち、フロントエンドを取り換えて、「小さな言語」を実装することができるようになります。また、Perl自身も、lexerなどはある程度Perlで記述できるようにしたいです。背後のモデルが違うので簡単ではないが、極端な話をすれば、「use ruby;」でRubyの文法を受け付けるフロントエンドというのもありえます。明示的に宣言するのであれば、極端な変化も受け入れられると考えています。
まつもと：フロントエンドを取り替えられることを目玉のひとつにしていたGNUのスクリプト言語Guileが、結局その部分は使われていないこととかを見ても、メタ言語的アプローチは成功していないんじゃないでしょうか。



写真3　オライリーブースでのサイン会。長蛇の列ができています。実はこの列の影にLarryさんが……


写真4　まつもとさんとLarryさん

全体的に非常におもしろいセッションでした。まつもとさんが、「世界的プログラマー」であるLarry氏と対等（？）に対話しているようすは、同じ日本人として少々誇らしくありました。しかし、今後は英語が重要なんですか

Rubyにおける新しいGCの実装

広島市立大学の木山真人さんが、Rubyの新バージョンに取り込まれる予定の世代別GC（ガベージコレクション）の実装について紹介するBOFでした。GCの実装がよくわかるプレゼンテーションでした（**写真5**）。

現在のRubyのGCは、すべてのオブジェクトをスキャンして、そのオブジェクトがまだ使われているかどうか判断する「マーク・アンド・スイープ」と呼ばれる方式を採用しています。しかし、この方式では大量のオブジェクトが発生したときに、処理が極端に遅くなるという欠点があります。木山さんが実装した「世代別GC」は、経験的に、オブジェクトの寿命は非常に短いものと、非常に長いものに分かれる傾向があることを利用して、寿命の長いものを別に管理し、ふだんは寿命の短いもの（正確には、短そうなもの）だけをスキャンすることで、GCの処理時間を短縮するものなのだそうです。

木山さんは、Rubyに世代別GCを実装することをテーマに修士論文を書き、Ruby 1.4のときにはかなり性能が改善されたそうです。しかし、現在のバージョン1.6では、本体でのGCの改善があったので、なかなかそれ以上の性能が出ないと嘆いていました。GCというものは、それだけで1冊の本が出るくらい奥の深いテーマだそうで、性能の良い実装も難しいのでしょう。

会場では、「PerlやPythonのようなリファレンスカウントを、選択的に使えるようにはならないか」という質問に対して、まつもとさんが「リファレンスカウントには受け入れられない欠点があるから、絶対に採用しない」と答えたりしていました。また、GCは裏方であるべきで、プログラムからGCに対してヒントを出すというやり方は嫌い

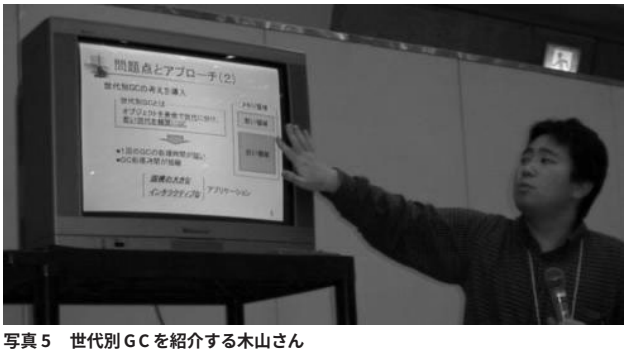


写真5　世代別GCを紹介する木山さん

基調講演：わが人生にRubyあり

まつもとさんによるキーノートスピーチです（**写真6**）。

なかなかおおげさなタイトルですが、内容は以前にも聞いたことのあるような、Rubyの設計思想の話が主でした。要旨は以下のとおりです。

- **Rubyはプログラマー（まつもとさん）の夢の結晶で、その基本的な動機は「作りたいから」であった。この点で、多くは特定の対象領域を持つ、他の言語とは異なる**
- **Rubyの目的は「ハッピーになること」と「自由」、そして「人類への貢献」である**
- **Rubyの設計原理は、単純さと複雑さのバランスにある。人間の心は複雑だが、複雑すぎるものは取り扱えない**
- **Rubyは「何ができるか」よりも「如何にできるか」に注目している。ソフトウェアが目的を果たすのはあたりまえ**
- **Rubyはオープンエンドである。機能を自然に追加でき、追加した機能をモジュール化できる。組み込み機能と追加した機能との間で差別はない**
- **今後は高速化と国際化をめざす。このために、IPAの補助金を受けている（この間AERAに載った件ですね）**
- **キラーアプリケーション募集中**

また、Rubyがまつもとさんの人生に与えた影響として



写真6　わが人生にRubyあり

は、転職した、講演するようになった、サイン会が開かれるようになった、恥ずかしい思いをした、海外旅行が決まった、などがあるそうです。

講演時間は90分だったのですが、1時間ほどで話は終わってしまい、後半30分弱は質問タイムでした。正直言って、あまり質問が殺到と言う感じではなかったのですが、「Rubyのほかに作ってみたいプログラムはなんですか」という質問に、「データベースやGUIライブラリ、ウィンドウシステムを作ってみたい」と答えてました。びっくり。もっとも「これらは、現在では1人の手には余りますけどね」と本気で作ろうというわけではなかったようですが。

世界制覇とかの話が出ると、「今で結構ハッピー、現状に満足している」と全然野望のないまつもとさんでした。なんだか気が抜ける口調ですが、まあ押し付けるよりも、良いものは良いとして、結果的に広まるというのが望みなのかもしれません。

ところで、来年1月終わりにパリで開かれる「Linux Expo Paris」に、まつもとさんは招待されたのだそうです。これまた、びっくり。今回のカンファレンスを含めて、外国人が日本に来ることはたびたびあるのですが、日本人が海外から招待されるなんて、これはすごいことではないかと思います。日本はコンピュータ業界のブラックホールだと言われる現状を、ぜひ打開してもらいたいと思います。これもレポートしたいんですが、ちょっとフランスまでの

WWWサーバサイドプログラミング

ApacheにRubyインタプリタを組み込むモジュールmod_rubyと、HTMLにRubyプログラムを埋め込むeRubyの作者であり、ゼータビッツ株式会社研究員の前田修吾さんによるWWWサーバサイドプログラミングの解説です。

CGIプログラミングには、以下の問題点があることが紹介されました。

・実行効率が悪い

CGIプログラムはリクエストのたびに起動されるが、プロセス起動のコストは無視できない

・HTMLを記述しにくい、読みにくい

CGIプログラム中にHTMLの断片が分散する

・セッション管理が面倒

HTTPは、状態がないプロトコルである

・セキュリティを確保しにくい

誤って外部からの入力を危険な処理に用いてしまうかもしれない

それに対するRuby的な解決策は、以下のとおりです。

・実行効率が悪い

mod_rubyで、Rubyインタプリタをサーバに組み込むことで、毎回プログラムが起動されることを回避する

・HTMLを記述しにくい、読みにくい

eRubyで、逆にRubyプログラムをHTMLに埋め込む。eRubyのプログラムでは、「<%」で始まり、「%>」で終わるブロックがRubyプログラムであり、その外側はそのまま出力される

・セッション管理が面倒

セッション管理を自動で行うCGI::Sessionライブラリを使う。CGI::Sessionライブラリは、セッション情報をクッキーとhiddenフィールドを併用して管理してくれる

・セキュリティを確保しにくい

Rubyが持つセキュリティレベル機能によって、実行できる操作を制限する。セキュリティレベルを適切に設定すれば、危険な操作を制限できる

そして、mod_rubyとeRubyのインストールや設定の方法が解説され、これらを使った例として、IMAPを使った電子メールリーダが紹介されました。このメールリーダ

ケーススタディ：Rubyで数値処理 ― NumRubyへ向けて

気象などの地球観測によって発生する大量のデータ（1日あたりTバイト単位で発生するのだそうです）を、Rubyで処理するという話です。インタプリタ言語ゆえに遅い、というデメリットを克服し、Rubyのような拡張性に優れ、柔軟性が高いフリーソフトウェアを利用することの実現を狙っているのだそうです。具体的には、最も速度に影響する部分の拡張ライブラリ化によって実現しているのですが、FORTRANで記述されたライブラリをリンクする手段を開発したり、オブジェクトの生成をできるだけ回避するなど、さまざまな工夫をしているようです。

次のセッション時間の関係で、前半の一部しか聞けなかったのですが、「Rubyには向かない」と言われていた数値処理の領域に、真っ向から挑戦する姿にちょっとだけ感動

オープンソースとビジネス パネルディスカッション

このセッションは、直接Rubyとは関係ないのですが、まつもとさんが登場するので参加してきました（**写真7**）。

自社開発のソフトウェアはすべてオープンソースにするという株式会社グッデイの前田さん、UPSのドライバをオープンソース化したことで売り上げが伸びたという三菱電機株式会社の藤巻さん、コミュニティは大切にしたいが、会社としての立場は複雑だという、FreeWnnの開発にもかかわられたオムロンソフトウェア株式会社の中西さん、Rubyの開発者が所属する会社（ゼータビッツ株式会社）というブランドの形成や技術力の証明にもなっているというまつもとさん、のそれぞれが自分の立場から意見を述べていました。

印象としては、もうオープンソースでビジネスになるのはあたりまえで、今後は競争にどう勝ち残るかという次のフェーズに入っていくように感じました。また、オープンソースにしただけでコミュニティができて、サポートのコストが要らなくなるわけではないという、あたりまえといえばあたりまえのことも再認識できました。

会場からは「大きい会社は事業部ごとの独立採算制をとっていることが多いので、オープンソースで稼がなくても、トータルでプラスならOKとはなりにくい」とか、「それでは取り残されるだけ」というような意見も出ていました。また、「オープンソースを含めたトータルなシステムで収益をあげるという安易なモデルで満足するのではなく、オープンソースそのものをビジネスにすることをまだ模索したい」という強い希望も聞かれました。

また、オープンソースソフトウェアを開発しているために冷遇されている人もいるという声に対して、まつもとさんは「そのソフトウェアをひっさげて、もっと理解ある会社に転職してはどうでしょうか。今なら理解のある会社がいくらでもあります。たとえばうち（ゼータビッツ）とかはどうでしょう？」と答えていました。司会の佐渡さんが



写真7　パネルディスカッション。左から前田さん、藤巻さん、中西さん、まつもとさん、佐渡さん

ワンライナーのすすめ+ Perl/Rubyの鉄人

ワンライナーの紹介と、Perl/Rubyの鉄人が与えられたテーマに沿って、その場でワンライナーを作ってみせるという企画です。まず、最初にTokyo Perl Mongersの前田薫さんがワンライナーについて紹介しました。

ワンライナーとは、書き捨てされることの多い1行程度のプログラムで、ほとんどの場合コマンドラインオプションで指定されます。PerlやRubyには、ワンライナーを支援するオプションや機能が提供されています。ワンライナーには以下のような利点があります。

- **コマンドラインでプログラムを書いてしまえる**
- **手早く仕事ができる**
- **バイナリデータに強い**
- **Perl/Rubyを使いこなしている実感がある**

3番目のは、必ずしもワンライナーの性質じゃないと思いますが。

その後、与えられたテーマに基づいて、各鉄人がワンライナーを作りました。鉄人は、Perl側がTokyo Perl Mongersの台場圭一さんとNamazuの高林哲さん、Ruby側がRed Hat Japanのやまだあきらさんと、yet another ruby porterのわたなべひろふみさんです。ところで、やまださんといい、わたなべさん、まつもとさんといい、なぜRuby関係者は名前をひらがなで名乗る人が多いのでしょう。一種の流行でしょうか？

テーマは以下のものが出されました。

(1) ファイル名.htmと.htmlが混在した環境で、全ファイル名を.htmlに揃える
(2) 与えられた数値を9進法で出力する
9進法の理由は、日曜の朝の番組を見るとわかるのだそうです
(3) 与えられた数を超えない素数を出力する
(4) Linux Conference 2000 Fallチュートリアル一覧（http://lc.linux.or.jp/t-detail.html）から、チュートリアルのタイトルと時間を抽出する

司会のはずの前田さんや、解説のはずのまつもとさんまでプログラミングやデバッグにのめりこんだり、会場からシェルやschemeを使った解答も飛び出すなど、終止笑いの絶えないセッションでした。まつもとさんによるPerlプログラムの解説などは、なかなか目にすることのできないものだったかもしれません。

ところで、上記のテーマ、あなたはワンライナーで書けますか？　**リスト1**に会場で見た答を紹介しておきます。紹介するのは一部のみです。当然Perlを使った解答もあり

Ruby開発者と話そう

まつもとさんと直接対話するための時間です。決して多い参加者とは言えませんでしたが、なかなかフレンドリーな時間でした。

■個人的な質問

「Larryの言語学のように、コンピュータ以外の経験がRubyの開発に役立ったことはあるか」という質問に対して、答は「ない。Rubyのサポートには、以前キリスト教の伝道師だった経験が役立ってるかも」ということでした。意外な経歴ですね。

■Rubyへの文句

以下の2点がまつもとさん自身から紹介されました。

- **ローカル変数のブロックスコープ（よく間違える）**
- **bang（!）メソッドの返値（よくひっかかる人がいる）**

会場からこれらに対して「こうしてはどうか」というアイデアも出ていました。そのほか、dRubyの実装で困っている点や、RWikiをどうするかなどといった具体的な話題も登場し、参加者同士で交渉と議論が行われるひとコマもありました。

■Rubyへの提案

「AS/400への移植を考えたいが、あまり拡大したくないのでは」という質問に対して、「そんなことはない。増えて困るのは『お客さん』だけ。開発者の拡大は大歓迎」とのことでした。

まつもとさんからは、「今は私がいろいろやりすぎている。なんでもはできないので、今後はやることを中心部の開発に限定していきたい」とのことでした。たしかに、ドキュメントやWebマスターまでもが、まつもとさんに集中しているのが現実です。そろそろ自発的な参加が求められているのでしょうか。それに応えてか、会場ではWin32版の新メンテナの立候補などの動きもあったようです。

そのほかにも、ファイルスコープやCの関数staticに相当する変数がほしいとか、Marshalをユーザーレベルで書ける機能を追加したいなどの技術的な提案がいくつも出ていました。

■今後の計画

キーノートにもあったとおり、近い将来としては高速化と多言語化（国際化よりもこちらのほうが適切な用語なのだそうです）を行うということでした。多言語化については、Muleの開発者から経験を聞いたりして、参考にしていくとのことでした。日本のお家芸でもある国際化、多言語化の成果として、すばらしいものを期待しましょう。

結局、セッションは少々時間を超過するほどになりまし

基調講演：Camel Lot ＃6

Perlの開発者Larry Wallさんによるキーノートスピー

▼リスト1　ワンライナー解答

```
(1) 全ファイル名を.htmlに揃える
ruby -e 'Dir["**/*.htm"].each {|i| File.rename i,
i+"l"}'
# Rubyのzsh風ワイルドカード機能を使ってます

(2) 与えられた数値を9進法で出力する
ruby -e 'n=81; d=[]; begin d.unshift(n%9); n/=9; end
while n != 0; p d.to_s'
# unshiftがミソ

guile> (number->string 81 9)
# 高林さんの解答。guileはschemeインタプリタです

echo 'obase=9;100' | bc
# 伝統的ツールを使った解答

(3) 与えられた数を超えない素数を出力する
ruby sample/sieve.rb 1000
# まつもとさんの解答。ワンライナーじゃありませんね（笑）

(4) は長いので割愛
```

チは、ほぼ満員の会場で同時通訳付きで行われました。「C言語とシェルしかない世界から、その隙間を埋めるためにPerlは誕生した」のだそうです。そして、「コンピュータサイエンス、言語学、常識、芸術」から良いものを取り込んできたそうです。それから、「ディズニーの言葉のように『夢見たことはなんでも実現できる』わけではないが、『夢見ないことは実現できない』というのは真実だ」と、夢つまり未来にあたるPerl 6について語りました。

Perl 6では、すべてを「再発明」するのだそうです。Perl 6がどのようなものになるかどうかは、ワーキンググループ形式で検討され、RFC（Request For Comment、コメント募集）という形で文書化されています。一種の規格のように見なされることの多いIANAのRFCと違って、これは「本物の」RFCなのだそうです。現在、361個のRFCが提案され、Larryさんが「これは良い、これは悪い、これは醜い」と検討、分類している途中です。分類中のRFCのいくつかが紹介されました。Perl 6は低レベル、高レベル、またメタ言語としても使えるものになるだろうとのことでした。今回Larryさんが紹介した、Perl 6の予想される特長を以下に示します。これらのうちのいくつかは、Rubyですでに実現されていますね。

- **非本質的な組み込み機能をモジュール化する**
- **全変数をオブジェクトとして用いることができる**
- **全変数は内部的にオブジェクトになる（vtable）**
- **「奇妙な変数」はオブジェクトかスコープの属性になる**
- **ファイルハンドルはオブジェクトになる**
- **オブジェクト参照の構文解析ルールが単純化される**
- **subscripting、slicingなどの構文解析ルールが単純化される**
- **do、evalなどの多義性が解決される**
- **型グロブはなくなる**
- **XSはなくなる。インターフェイスは低レベルPerlで記述される**
- **オブジェクト属性宣言方法が標準化される**
- **オブジェクト属性アクセス方法がアクセサによって標準化される**
- **真のガベージコレクション**
- **オブジェクト指向例外処理**
- **より簡単な式展開**
- **演算子優先順位が単純化されるかもしれない**

今回のプレゼン資料とほぼ同じものが、http://dev.perl.org/~ask/als/talk.htmlで参照できます。今回の話を聞いて感じたのは、Larry Wallさんとまつもとさんの類似性でした。いや、外見は全然似てないんですが（笑）。

「対談：PerlとRubyの比較」でも感じたのですが、この2人は思想的にかなり似ていると思います。単にUNIX文化の影響を受けているというだけでなく、言語を通じて人間の仕事を支援しようというか、ハッピーになることをめざすというか……。もちろん、まつもとさんは、Larryさんの思想に影響を受けたことを公言していますから、当然なのかもしれませんが、ユーモア好きなところと、ちっとも偉ぶらないところも似てますね。

過去の歴史と資産を抱えつつ、未来をめざすPerl。新しい言語としてアドバンテージを持ちつつ、Perlを追いかけ

謝辞

今回、筆者のデジカメの不調などから、JLA理事のやまだあきらさん（Red Hat Japan所属）から画像の多くを提供していただきました。感謝いたします。また、今回講師やスタッフとして参加された皆さんにも感謝いたしま

Column

読者プレゼント

今回は、筆者がゲットしたまつもとさんのサイン入り『Rubyデスクトップリファレンス』（写真8、写真9）を1名様にプレゼントします。ご希望の方は、本誌168ページのプレゼント応募方法にしたがって応募してください。



写真8　Rubyデスクトップリファレンス



写真9　まつもとさんのサインがバッチリ

[超]入門シェルスクリプト



Linuxの標準シェルであるbashのシェルスクリプトについて学ぶ本連載。今回は、コマンドの実行結果によって分岐を行うif制御構造や、終了ステータスを指定してスクリプトを終了する組み込みコマンドexitなどについて説明した後、指定したコマンドの種類を判定するシェルスクリプトを作成する。

第4回　if制御構造を使った分岐処理（その1）

文：大池浩一
Text:Koichi Oike

前回紹介したfor制御構造を利用すると、シェルスクリプト内で同じような処理を繰り返し行える。処理の流れを変えるこうした「フロー制御構造」のもうひとつの柱が、指定した条件が成立するかどうかにより異なる処理を行う分岐処理だ。

世の中で使われているシェルスクリプトのほとんどには、「指定したコマンドが正常に終了した場合にだけ処理を続ける」とか、「指定したファイルが存在している場合にだけ特定の処理を行う」といった分岐が含まれている。これにより、コマンドラインから直接実行するのは難しい複雑な処理を自動的に判断して行う「賢い」スクリプトを作ることができるのだ。

たとえば、ソフトのビルドとインストールを行うときのように、一連のコマンドを続けて実行する処理をシェルスクリプト化することを考えてみよう。人間がコマンドをひとつひとつコマンドラインで入力する場合は、途中のコマンドでエラーが発生したのを確認して、次のコマンドの実行を中止できる。

一方、それぞれのコマンドを単にシェルスクリプトに列挙すると、あるコマンドでエラーが起きても、次のコマンドをおかまいなしにそのまま実行してしまう。しかし、今回紹介するif制御構造を使ったシェルスクリプトならば、エラーが起きたらそれ以降のコマンドは実行しないといった分岐処理を行えるわけだ。なお、さまざまな条件判断の方法については次回取り上げる。

分岐を実現するif制御構造

Linuxの標準シェルであるbashや、そのベースとなったBourneシェルでは、分岐処理を行うif、caseの2つの制御構造が用意されている。今回は、このうちのif制御構造について取り上げる。

CやBASICなど他のプログラミング言語の経験がある人は、シェルスクリプトのifはかなり勝手が違うので気をつけてほしい。たいていのプログラミング言語では、ifは「指定された条件が真（成立）か、偽（不成立）か」という条件式の真偽値に基づいて分岐を行う。

これに対し、シェルスクリプトのifは、「指定されたコマンドが正常終了したか、異常終了したか」という終了ステータスに基づいて分岐を行うのだ。なお、条件式の真偽値に基づいて分岐を行うには、与えられた条件式の真偽値を終了ステータスに反映させるコマンド「test」（あるいは「[」）を利用する（詳細は次回）。

以下では、

- **if制御構造の記述のしかた**
- **スクリプトを途中で終了する方法**

などの説明を行った後、実際にif制御構造を利用したスクリプトの作成を行う。

●if制御構造を記述する

シェルスクリプトのif制御構造の書式にはいくつかのパターンがある。まずはもっとも簡単な書式から始めて、しだいに複雑な書式へ発展させることにしよう。

(1)コマンドが正常終了した場合の処理を行う

if制御構造のもっとも単純な書式は次のようになる。

```
if コマンド
then
  コマンドが正常終了した場合に実行する内容
fi
```

以下の説明では、コマンドが正常終了した場合に実行する内容の前に書く「then」を、「if」と同じ行にまとめて書くことにする。それには、コマンドの後ろに文の区切りを表す「;」（セミコロン）をつけて、

```
if コマンド; then
```

とすればいい。

それでは内容を説明していこう。まず、「コマンド」には、コマンドラインと同じように、コマンドとそのオプション、ファイル名などをそのまま書けばいい。

```
if grep 'hoge' *.txt; then
```

といった具合だ。ワイルドカード（＊、？など）を利用したファイルの展開もちゃんと行われる。

実行したコマンドが正常終了したかどうかは、コマンドの「終了ステータス」で判断される。終了ステータスとは、コマンドが終了する際に呼び出し元（ここではシェル）に返す値のことだ。この値は0～255の範囲の整数で、通常は0が正常終了、1～255が異常終了を意味する。

異常終了の終了ステータスに幅広い値が割り当てられているのは、「異常」の具体的な内容を値で区別するためだ。ただし、今回説明する範囲では、「終了ステータスが0（正常終了）か、そうでない（異常終了）か」という単純な分岐しか行えない。終了ステータスの値を整数値と比較して、それぞれの状況に応じた処理を行うには、コマンド「test」（あるいは「[」）を利用する必要がある。詳しくは次回説明することにしよう。

さて、コマンドが正常終了した場合は、「then」と「fi」（「if」を逆向きに綴ったもの）の間に記述された内容が実行される。この部分には複数のコマンドラインを記述可能だ。一方、コマンドが異常終了した場合は、「then」と「fi」の間の内容はスキップされ、「fi」の後の処理に移る。ちなみに、終了部に「fi」という逆綴り単語が使われるのは、bashなどのBourne系シェルの一種の「クセ」だ（詳しくはコラムを参照）。

(2)コマンドが異常終了した場合の処理を追加する

if制御構造のもっとも単純な書式では、「コマンドが異常



図　if/else制御構造による分岐処理

終了した場合にエラーメッセージを表示する」といった処理を記述できない。こうした処理を行うために、if制御構造には次の書式が用意されている。

```
if コマンド; then
  コマンドが正常終了した場合に実行する内容
else
  コマンドが異常終了した場合に実行する内容
fi
```

さきほどと違うのは、コマンドが異常終了した場合に実行する内容を、「else」と「fi」の間に記述できることだ。一方、コマンドが正常終了した場合に実行する内容は「then」と「else」の間に記述することになる。その他の点は(1)とまったく同じだ。

簡単な例題として、カレントディレクトリにあるCのソースファイルに、「gtk_set_locale」という単語が含まれていれば「OK」と表示するスクリプトを作ってみよう（これは、GTK＋を利用して作られたプログラムに、日本語などのロケール処理を行う関数gtk_set_localeが含まれているか調べるためのものだ）。

検索用のコマンドとしてはgrepを使用する。一般に、Cのソースファイルはワイルドカードを使って「＊.c」と書けるから、これにマッチしたファイルに「gtk_set_locale」が含まれているかどうか調べればいい。よって、スクリプトの内容は次のようになる。

```
#!/bin/sh
if grep -qs 'gtk_set_locale' *.c; then
  echo OK
fi
```

まず、if制御構造のコマンド部分

```
grep -qs 'gtk_set_locale' *.c
```

が実行される。「＊.c」が具体的なファイル名に展開されてgrepに渡され、それらに対してパターン「gtk_set_locale」が検索される。通常、grepはパターンを含む行を見つけるとそれを出力するし、指定されたファイルが見つからない場合はエラーメッセージを出力するが、ここでは-qと-sオプションによりどちらも抑制される。

grepが終了すると、終了ステータスに応じて分岐が行われる。パターンが含まれる場合は正常終了（終了ステータス0）なので、「echo OK」が実行されて「OK」と表示される。

一方、パターンにマッチしなかった場合（同1）、Cのソースファイル自体が見つからない場合など（同2）、あるいはgrep自体が実行できなかった場合（同127）は、いずれも異常終了なので、「then」と「fi」の間はスキップされ、何も表示しないまま終了する。

さらに、異常終了した場合には「NG」と表示するには、次のように書けばいい。

```
#!/bin/sh
if grep -qs 'gtk_set_locale' *.c; then
  echo OK
else
  echo NG
fi
```

grepが正常終了した場合は、「then」と「else」の間の内容（ここでは「echo OK」）が実行され、「else」から「fi」の間の処理はスキップされる。一方、異常終了した場合は、「then」から「else」の間の処理はスキップされ、「else」から「fi」の間の内容（ここでは「echo NG」）が実行されることになる（**図**）。

Column

では聞くが、fiやesacとは一体何なのだ！

本文中でも触れているように、if制御構造のブロックの終わりを示すために使われる「fi」は、「if」の綴りを逆さにしたものだ。同様に、複数分岐を行うcase制御構造でもブロックの終わりに逆綴り単語の「esac」が使われる。こうした見なれない逆綴り単語を使うのは、Bourne系シェルに独特の「クセ」だ。tcshなどのCsh系のシェルでは、制御構造のブロックの終わりには「endif」や「endsw」などのわかりやすい単語が使われる。

その他のスクリプト系言語では、PerlやAWKのようにブロックを中括弧「{」と「}」で囲むものや、Rubyのように一貫して「end」を使うもの（中括弧も使用可）、Pythonのようにインデント（字下げ）でブロックを表すものなど、バリエーションに富んでいる。

幸いなことに、Bourne系シェルでも、繰り返し制御構造（for/while/untilの3種類）のほうは、ブロックの始まりは「do」、終わりは「done」で統一されている。「rof」とか「elihw」、「litnu」なんて単語は登場しないので海原先生も安心だ。

(3)複数のコマンドを次々に実行して分岐する

if制御構造のもっとも複雑な処理は、最初のコマンドが異常終了した場合には別のコマンドを実行し、また異常終了した場合にはさらに別のコマンドを実行…といった具合に、複数の分岐処理を連続して行うものだ。

書式は次のようになる。

```
if コマンドA; then
  コマンドAが正常終了した場合に実行する内容
elif コマンドB; then
  コマンドBが正常終了した場合に実行する内容
else
  コマンドBが異常終了した場合に実行する内容
fi
```

ここでは「elif」を1つしか記述していないが、実際には何個続けて書いてもよい。また、C言語などでは「else if」と書くところを、シェルスクリプトでは「elif」と1語で書く点にも注意してほしい。

さきほどと違うのは、最初のコマンド（コマンドA）が異常終了した場合に、「elif」によって別のコマンド（コマンドB）が実行される点だ。なお、コマンドAが正常終了した場合は、「elif」から「fi」までの処理はスキップされるので、コマンドBは実行されない。

先の例題で、「gtk_set_locale」が見つからなかった場合の処理を変更してみよう。今度は、Cのソースファイルから「gtk_init」を探し、見つかった場合はファイル名と「contain(s) gtk_init」、見つからなかった場合は「NG」を表示することにする。

実際のスクリプトは以下のようになる。

```
#!/bin/sh
if grep -qs 'gtk_set_locale' *.c; then
  echo OK
elif grep -ls 'gtk_init' *.c; then
  echo 'contain(s) gtk_init'
else
  echo NG
fi
```

まず、ifの後のコマンド、

```
grep -qs 'gtk_set_locale' *.c
```

により、パターン「gtk_set_locale」が検索される。

パターンにマッチした行を持つファイルがあれば、「echo OK」が実行され、「elif」から「fi」までの処理はすべてスキップされる。

パターンにマッチしないなどの理由で異常終了した場合は、elifの後のコマンド、

```
grep -ls 'gtk_init' *.c
```

が実行され、パターン「gtk_init」が検索される。

パターンにマッチする行を持つファイルがあれば、-lオプションによりそのファイル名だけが表示され、「echo 'contain(s) gtk_init'」が実行される（その後「fi」までの処理はスキップ）。

もし、パターンにマッチしないなどの理由で2つめのgrepも異常終了した場合は、「echo NG」が実行される。

この節での要点をまとめると、

- **if制御構造では、実行したコマンドの終了ステータスにより分岐する。**
- **異常終了時の処理はelse節で行う。**
- **異常終了時にさらに別のコマンドを実行して分岐するelif節も用意されている（複数指定可）。**

ということになる。

●スクリプトを途中で終了する組み込みコマンドexit

if制御構造では、コマンドの正常終了と異常終了によって分岐した処理が終わった後は、いずれも「fi」の後に書かれた内容が（もしあれば）実行される。

しかし、異常終了したときには「fi」の後に続く処理を行いたくない場合も考えられる。このようなときには、スクリプトを途中で終了する組み込みコマンドexitを使うといい。

書式は次のようになる。

```
exit 終了ステータス（省略可）
```

「終了ステータス」には、シェルスクリプトが呼び出し元（通常はシェル）に返す値を指定する。他のコマンドと同様に、正常終了の場合は0、異常終了の場合は1～255の範囲で適当に意味を決めて設定すればいい。

一方、exitを使わずにスクリプトが終了する場合や、

exitで終了ステータスの指定を省略した場合は、「直前に実行したコマンドの終了ステータス」がスクリプトの終了ステータスとして返される。

たとえば、

```
#!/bin/sh
grep 'gtk_set_locale' *.c
```

というスクリプトの終了ステータスには、grepの終了ステータスがそのまま使われる。指定されたパターンが見つからないでgrepが異常終了した場合（終了ステータス1）、スクリプトの終了ステータスも1になる。このように、実行するコマンドが1つだけのシェルスクリプトでは、終了ステータスを気にする必要はない。

しかし、終了ステータスを利用したいコマンドを実行した後で、別のコマンドを実行せざるをえない場合もある。こうしたときには、直前に実行したコマンドの終了ステータスが設定される組み込み変数「?」（参照形は「$?」）を、いったん別のシェル変数に保存し、exitの引数で参照すればいい。

たとえば、次のように書く。

```
if コマンド; then
  コマンドが正常終了した場合に実行する内容
else
  es=$?
  コマンドが異常終了した場合に実行する内容
  exit $es
fi
```

コマンドが異常終了すると、まずは「$?」で参照できるgrepの終了ステータスをシェル変数esに保存しておく。それ以降に実行する内容によって「$?」の値は再設定されてしまうが、分岐処理の最後に「exit $es」とすることで、保存した終了ステータスをスクリプトの終了ステータスとして利用して実行を終了できる。

このほか、Bourneシェルにはないbash独自の特徴として、if制御構造の終了ステータス判定を反転する「！」がある。「！」をコマンドの前に指定すると、異常終了した場合の処理を「then」以下に書けるというものだ（コマンドと「！」の間には1個以上のスペースを入れること）。

```
if ! コマンド; then
  異常終了した場合に実行する内容
fi
```

この書き方はエラー処理を行うのに便利なので覚えておくといい。「then」と「fi」の間にエラーメッセージなどを書いて、最後にexitでシェルスクリプトの実行を終了するのが一般的だ。これで、「fi」以降に正常終了した場合の処理をそのまま書くことができる。

ただし、「！」による終了ステータス判定の反転機能はBourneシェルには含まれていないので気をつけよう。利用する場合は、スクリプトの1行目の「#!」の後に書くシェルを、Bourneシェルを意味する「/bin/sh」ではなく、bashを意味する「/bin/bash」にするといい。

もっとも、Linuxにおける「/bin/sh」は、実際には「/bin/bash」のシンボリックリンクなので、どちらの場合もbashが実行される。

この節での要点をまとめると、

- **スクリプトの途中で実行を終了するにはexitを使う。引数にはスクリプトの終了ステータスを指定。**
- **終了ステータスを省略すると、直前に実行したコマンドの終了ステータスが使われる。**
- **直前に実行したコマンドの終了ステータスは「$?」で参照できる。**
- **bashでは、if制御構造のコマンドの前に「!」を付けることで終了ステータス判定を反転できる。**

Column

timeっていうスクリプトが動かないんですが…

for制御構造で使われるfor／in／do／done、if制御構造で使われるif／then／elif／else／fiは、Bourbe系シェルでは特別な意味を持つ「予約語」だ。

予約語による制御構造などの処理は、通常のコマンドの検索・実行処理よりもずっと前の段階で行われる。このため、予約語と同名のシェルスクリプトをPATHに設定されたディレクトリに置いても、通常の方法では決して実行できない（「./for」など、ディレクトリ付きで指定すれば実行可能）。

英単語の予約語には、このほかにもcase／esac／while／until／function／select／timeがある。selectやtimeなど、ついスクリプト名に使ってしまいそうなものも含まれるので気をつけよう。

ということになる。

今月のスクリプト

後半は、既存のスクリプトの内容を解説する「スクリプトを読む」と、与えられたテーマを実現するスクリプトの作成手順を説明する「スクリプトを書く」で構成する。今月は紙幅の都合上「スクリプトを読む」はお休みする。「スクリプトを書く」では、

- **指定したコマンド（複数可）のファイルの種類を判定する「cmdfile」**

を作成する。

今回説明したif制御構造はもちろん、コマンド置換やシェル変数、for制御構造など以前の連載で説明した内容を含んでいるので、不明な点がある場合は第2回と第3回の記事も参照してほしい。

●指定したコマンドの種類を判定するには

UNIX系OSで実行可能なコマンドには、C言語などで作られた実行可能なプログラムファイルのほか、Bourne系シェルやCシェル、あるいはスクリプト言語（Perl、AWK、Rubyなど）で書かれたスクリプト、別のコマンドへのシンボリックリンクなどさまざまな種類のファイルがある。

このようなコマンドの種類を判定するには、ファイルの種類を判定するコマンドfileを使うと便利だ。fileは、引数で指定されたファイル（複数可）の種類を、

- **ファイルシステムテスト…スペシャルファイル（シンボリックリンクなど）かどうか調べる。**
- **マジックナンバーテスト…ファイル先頭のマジックナンバー（スクリプトの「#!」など）を調べる。**
- **言語テスト…テキストファイルの場合は、そこで使われている文字列を調べる。**

という3段階のテストを通じて調べ、結果を標準出力（通常は端末画面）に出力してくれる。たとえば、

```
$ file hoge hoge.txt hoge.mp3
hoge:          Bourne shell script text
hoge.txt:      EUC text
hoge.mp3:      MP3, 128 kBits, 44.1 kHz, JStereo
```

といった具合だ。シェルスクリプトではシェルの系統、テキストファイルは文字コード、MP3ファイルはビットレートや周波数といった具合に、内容に関するかなり細かい情報まで出力される。

fileはコマンドを特別扱いしないで一般のファイルと同様に扱う。つまり、コマンドの種類を調べるときには、コマンド名だけ指定したのではだめで、ディレクトリ付きのフルパス（あるいは相対パス）で指定する必要がある。たとえば、grepは/binに置かれているので、

```
$ file /bin/grep
/bin/grep: ELF 32-ビット LSB executable, …
```

とすればいい。

私たちが使うコマンドは、/binのほかにも/usr/bin、/usr/local/bin、~/binなどさまざまなディレクトリに散在しているため、まずはどのディレクトリにコマンドが置かれているか調べる必要がある。それには、指定したコマンドのフルパスを標準出力（通常は端末画面）に出力するコマンドwhichを使うといい。

たとえば、grepの場合は、

```
$ which grep
/bin/grep
```

といった具合に表示される。

本連載の第2回に登場した「コマンド置換」(コマンドを「`」（逆引用符）で囲む)を利用すれば、whichの出力であるコマンドのフルパスを、そのままfileの引数として利用できる。

grepを例にとると、

```
$ file `which grep`
/bin/grep: ELF 32-ビット LSB executable, …
```

となる。

whichもfileも複数の引数に対応しているので、コマンド名は複数指定でき、

```
$ file `which grep zless mkdic`
/bin/grep:     ELF 32-ビット LSB executable, …
```

```
/usr/bin/zless: Bourne shell script text
/usr/bin/mkdic: symbolic link to catdic
```

といった具合に、それぞれのコマンドのファイルの種類を調べて表示してくれる。

●スクリプト「cmdfile」を作成する

前節で作成した「file `which コマンド名…`」というコマンドラインを叩き台として、シェルスクリプト「cmdfile」を作成することにしよう。

(1)コマンドラインをそのままシェルスクリプト化

前節で説明したコマンドラインをそのままシェルスクリプトにすると、**リスト1**のようになる。

「"$@"」は前回詳しく説明したように、シェルスクリプト実行時のコマンドライン引数（ここでは調べたいコマンド名の並び）の内容をそれぞれ「"」でクォーティングし、スペース1個で区切って並べたものに展開される。

すると、whichがそれらのコマンドのファイル名をフルパスで出力し、コマンド置換により「`」で囲まれた部分が出力結果で置き換えられる。続いて、fileが各ファイルの種類を調べて出力する。

fileとwhichがどちらもフルパスで記述されているのは、実行時のシェルのPATHの内容や、エイリアス（別名）の定義に影響されずに、確実にコマンドを起動するためだ。

このスクリプトは、実際に存在するコマンドを指定する限りはちゃんと動作するし、複数のコマンドを指定した場合には、それぞれの種類が1行ずつ表示される。

問題は、コマンドラインでコマンドを1つも指定しない場合や、実際には存在しないコマンドを指定した場合に思わぬ結果が出力されることだ。実用的なスクリプトでは、こうしたケースでもユーザーが納得する動作をするようにエラー処理をする必要がある。

(2)コマンド名を1つも指定しない場合に対応する

cmdfileのコマンドラインで、コマンドを1つも指定しないで実行すると、なぜかwhichとfileの使用法（Usage）が端末画面に表示されてしまう（**画面1**）。まずはその原因を考えることから始めよう。

▼リスト1　スクリプト「cmdfile」の内容（第1版）

```
1: #!/bin/sh
2: /usr/bin/file `/usr/bin/which "$@"`
```

この場合、「"$@"」は何にも展開されないので、whichは引数なしで実行される。このため、whichはUsageを標準エラー出力（通常は端末画面）に出し、標準出力には何も出力しない。すると、コマンド置換の結果が空文字列になるので、fileも同様に引数なしで実行され、使い方が標準エラー出力に出力されるのだ。

原因がわかったところで解決方法を考えよう。典型的な解決方法は、if制御構造で条件判断を行うものだが、次回の内容なのでここでは扱わない。次善の策は、「whichが異常終了したらfileは実行しない」というものだ。whichが異常終了したかどうかはif制御構造で判断できる。

スクリプトは**リスト2**のようになる。まず、2行目のコマンド置換部分の、

```
/usr/bin/which "$@" 2> /dev/null
```

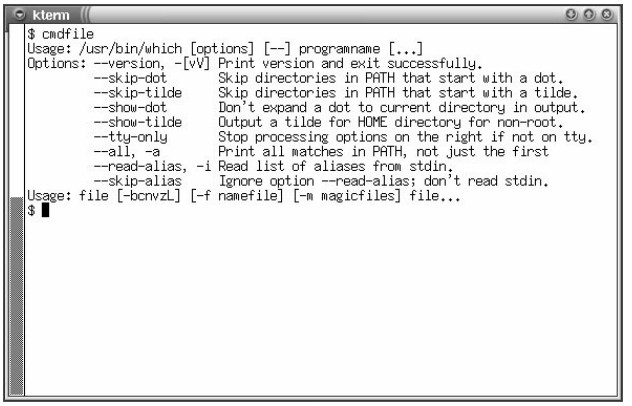
が実行され、出力結果がシェル変数listに格納される。「2>」は標準エラー出力のリダイレクトを意味しており、Usageを画面に表示させないために使う。リダイレクト先の「/dev/null」は何を書いても即座に捨てられるゴミ箱のようなデバイスだ。

whichが正常終了した場合は、if制御構造により「then」と「fi」の間にある、

```
/usr/bin/file $list
```

が実行される。fileの引数として、whichの出力結果が格納されたlistが変数展開されるので、このときの出力はリスト1のスクリプトと同じだ。

一方、whichが異常終了した場合は、「fi」までの処理がスキップされるので、fileは実行されず、結果として画面には何も表示されない。



画面1　whichとfileの使用法（Usage）が表示される

(3)指定したコマンドが存在しなかった場合に対応する

続いて、指定したコマンドが存在しなかった場合の処理を行う。まずは、存在しないコマンドのみ指定した場合について考えよう。

最初のスクリプト（**リスト1**）では、whichが「コマンドが見つからない」旨のエラーメッセージを表示し、コマンド置換の結果は空文字列になるので、fileはUsageを表示する（どちらの表示も標準エラー出力）。これに対し、改良を加えたスクリプト（**リスト2**）では、whichのエラーメッセージは画面に表示されず、if制御構造によりfileの実行もスキップされるため画面には何も表示されない。なんとか納得してもらえそうな動作だ。

次に、存在するコマンドと存在しないコマンドを混在させた場合について考えてみよう。

whichは、存在するコマンドについてはファイル名（フルパス）を標準出力に表示し、存在しないコマンドに対してはエラーメッセージを標準エラー出力に表示する。したがって、コマンド置換の結果は存在するコマンドのファイル名だけが並んだリストになり、シェル変数listに格納される。

ここで問題となるのはwhichの終了ステータスだ。存在するコマンドと存在しないコマンドが混在している場合、どのような終了ステータスを返すのだろうか。whichのマニュアルの「RETURN VALUE」の項を読むと、終了ステータスは存在しないコマンドの数と等しいことがわかる。1つでも存在しないファイルがあると、whichは異常終了の終了ステータスを返すわけだ。

このため、リスト2のスクリプトでは、存在しないコマンドが1つでも含まれているとfileの実行がスキップされてしまう。使う側からすれば、これはちょっと納得できない動作だろう。存在しないコマンドは「hogehogeは存在しません」とでもエラーメッセージを表示し、存在するものについては通常のfileによる表示を行うという動作が望ましいと思われる。

そこで、完成版のスクリプトでは、for制御構造を使ってコマンドライン引数を1つずつ扱うようにした（**リスト3**）。2行目のfor制御構造で使われるループ変数fには、コマンドライン引数で指定したコマンド名が繰り返しのたびに1つずつ入る。

3行目のif制御構造のコマンド部で実行されるwhichが、そのコマンド名をフルパスのファイル名に変換し、その結果がコマンド置換によりシェル変数cpathに格納される。whichが正常終了した場合は「then」以下の内容（4行目）が実行され、異常終了すると「else」以下の内容（6行目）が実行される。

このように、コマンドを1つずつ分離して扱うことにより、各コマンドの存在の有無を終了ステータスで判別して、分岐処理できるようになった。存在するコマンドと存在しないコマンドを混在させた場合の実際の出力を**画面2**に示す。

さらに付け加えるなら、スクリプト自体の終了ステータスを、存在しないコマンドの数に対応させたいところだが、それにはシェル変数の数値演算を行う必要があるので、今回は見送ることにする。

なお、世の中で使われているシェルスクリプトのほとんどは、if制御構造のコマンド部で「test」（または「[」）というコマンドを実行している。testを使うことで、ファイルの存在チェックやシェル変数と文字列の比較など、いろいろな条件判断を行えるのだ。このようなif制御構造とtestの組み合わせについては、次回詳しく説明することにしよう。

▼リスト2　スクリプト「cmdfile」の内容（第2版）

```
1: #!/bin/sh
2: if list=`/usr/bin/which "$@" 2> /dev/null`; then
3:   /usr/bin/file $list
4: fi
```

▼リスト3　スクリプト「cmdfile」の内容(完成版)

```
1: #!/bin/sh
2: for f in "$@"; do
3:   if cpath=`/usr/bin/which "$f" 2> /dev/null`; then
4:     /usr/bin/file "$cpath"
5:   else
6:     echo "$fは見つかりません" > /dev/stderr
7:   fi
8: done
```



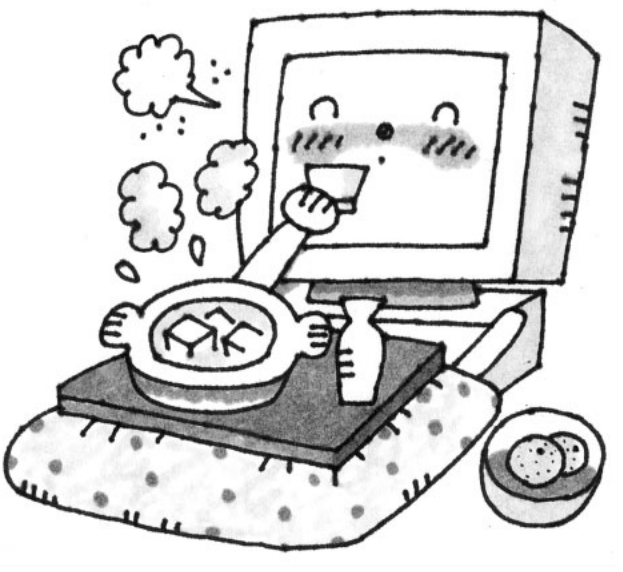
画面2　指定されたコマンドの種類を表示

いろんな人にやさしいEmacs

Emacsはじめました

第11回　喋るEmacsと翻訳するEmacs

今回は、お手元のEmacsに「喋らせる」「翻訳させる」という21世紀初頭を飾るにふさわしい2題で迫ってみましょう。ひとつは開発途中のソフト、もうひとつは商用ソフトなので、いつものように「ではインストールしてみましょう」とはしません。でも、こんなこともできるんだな、と見ていただくだけでも面白いと思います。



文：佐々木太良
Text : Taroh Sasaki

Illustration : Manami Kato

喋るEmacs

我らがEmacsはもともと文字ベースでしたが、近年ではGUIも採り入れられてきています。ならばAUI（聴覚ユーザーインターフェイス）があってもいいじゃないかと、Emacsの画面を読み上げてくれるEmacspeakというソフトウェアを提案・開発した人がいます（http://emacspeak.sourceforge.net/）。Emacsが「喋る」だけでも素敵なことですが、EmacspeakではカレンダーやC言語といった各アプリケーションや動作モードに合わせ、時としてユーモラスな方法で読み上げる方法を変えてくれます。そんな楽しい、オレのPCサイバー感100倍のEmacspeaksですが、マウスカーソルを見ることができない視覚障害者には、たいへん幸せな試みとなります。Emacspeakを作ったT.V.Raman博士も、実は視覚障害者なのだそうです。

視覚障害者にとってGUIの使用は困難ですから、むしろCUIのほうが都合がよいのです。足掛け2世紀にわたって使われてきたUNIX系のOSは、この点たいへん優れています（笑）。たとえばプログラミングをする場合、これまではよく、画面を読み上げる機能や点字ディスプレイへのインターフェイスソフトウェアが利用できるMS-DOSなどのパソコンを使い、telnetなどでリモートログインしてCUIを利用してきました。Emacspeakはこの方法の代替となりうるものです。

Emacspeakを使ってみると

Emacspeakは、見た目にはまったく普通のEmacsと同じで、喋っているようすを誌上で伝えるのは難しいのですが、付録のCD-ROMにデモンストレーションの音声を収録したので、そちらも聴いてみてください。おもしろいですよ。コラム「Emacspeakのしくみ」で触れているように、Emacspeakはelispアプリケーションごとに読み上げる方法を変えられます。ここで紹介している操作はその一例です。

■通常の動作（テキストモードなど）

C-n／**C-p**などでポイントを移動すると、その行を読み上げてくれます。また、**C-e s**で読み上げを停止できます。これは、以下に紹介するアプリケーションのモードでもかなりの部分が共通です。逆に言うと、アプリケーション別のアドバイスがなくても、それなりにスクリーン上のものを読み上げることはできるということです。

■音声フォントとオーディトリアイコン

いろいろなモードに共通の面白い特徴として、音声フォントによる声色の変化と、オーディトリアイコンというサイン（効果音）が挙げられます。Emacsではテキストに色などの属性をつけて、着目したい部分を強調表示できま

した。AUIでこれに相当するのが音声フォントで、属性ごとに少女の声やピッチの高低などを設定し、ユーザーの注意を引くことができます。

オーディトリアイコンというのは、要するに効果音です。状況に応じた効果的なオーディトリアイコンを使えば、喋っている内容がぐっと把握しやすくなります。

■eggを使ったかな漢字変換

かな漢字変換時にはいろいろなモードがあるので、今どの候補を選択しているかがわかるようにするには、表示されているものをただ読み上げるだけでは不十分です。そこで、石川准・静岡県立大学教授（http://www.u-shizuoka-ken.ac.jp/~ishikawa/）が開発した、グラスルーツというDOS用画面読みプログラムの辞書（詳細読み辞書）を利用して、変換モードで「亜」という候補が提示されていれば「あじあの『あ』」と喋るようになっています。

また、「ひらがなとカタカナ」を読み上げるときには「ヒラガナト（ぴっ）カタカナ」とするような拡張が検討されているそうです。

■カレンダーモード

アプリケーションごとに読み上げる方法を変えた効果は、このモードが顕著でしょう。カレンダーを「なな はち きゅう……」と読まれても、なんのことやらですが、Emacspeakのカレンダモードなら、「1月7日（日）」にカーソルを置くと“ Sunday, January 7th ”と読み上げてくれます。

■Diredモード

ファイルの一覧で「でぃーあーるだぶりゅーえっくす……」（drwxr-xr-x ...）と読み上げられても嬉しくありません。そこで、Diredモードでは必要なところだけ読んでくれます。ディレクトリ名はEmacsのDiredモードでも文字属性を変えて表示しているので、ファイルかディレクトリかは声色を聞けばわかります。また、diredモードで：を押すとファイルサイズ、/を押すとパーミッションなど、必要な部分だけ読み上げることもできます。

■w3モード

Emacsでは、HTML文書のタグやリンクは文字属性を変えて表示されますが、我らがEmacspeakは声色が変わるだけでなく、たとえば表なら縦一列に読む機能があって、普通のテキストとして読むのとは違った快適さです。

■Cモード

Cモードは、12月号でちょっと触れたように、C言語のプログラミング時に都合のよいように動作してくれるモードです。実用向けでつまらないと思いがちですが、どうしてEmacspeakのCモードは遊び心がいっぱいです。キーワードや関数名や文字列内などは独特の声になり、コメント行はアクセントのない日本語か英語で喋ります。コンパイル中にエラーが出ると、オーディトリアイコンで効果音が再生されるので、締め切り近いプログラマーはいっそう落ち込むかもしれません。デバッグ時にはエラー行を示し、エラー箇所にジャンプすると読み上げてくれます。正常にコンパイルが終わると電車の警笛音で知らせてくれ、その

Column

バイリンガルEmacspeakとBEP

BEP（Bilingual Emacspeak Project）は、英語しか話せないEmacspeakを日本語化しようという人たちの集まりで、メーリングリストによる活動をしています（http://www.argv.org/bep/）。

BEPはWindowsでもLinuxでも同じようにEmacsをバイリンガルで音声利用できるようにすることを目指していて、将来は、英語や日本語以外にも、複数言語を切り替えられるように拡張する予定だということです。

Emacspeakのオリジナル版は、英語版の音声合成エンジン（DECtalkなど）と組み合わせて英語を喋ります。では、日本語の音声合成ソフトを探してきて入れ替えればよいかというと、そう簡単にはいきません。まず第一に、日本語の音声合成エンジン自体がなかったのです。このためBEPは、クリエートシステム開発からLinux版音声合成SDKという音声合成ツールキットのライセンスを購入し、独自の音声合成エンジンを作っています。BEPの成果をなかなか公開できないのは、SDKの部分がフリーではなく、一般には入手しにくい、という事情があったようです。

次に日本語化されたEmacspeakであっても、英語の音声合成エンジンは依然として必要だという事情があります。Emacspeakを使う人は技術文書をよく使用すると思われますが、英語の部分の綴りをカタカナ英語で読まれると、わかりにくいからです。

BEPの開発部分はオープンソースですし、いずれ簡単インストールのためのパッケージも配布されるでしょう。現状ではEmacspeakの最新情報を得られるのはBEPのメーリングリストですから、お手伝いしたい人や、喋るEmacsのユーザーとしてフィードバックをしたい人は、BEPのメーリングリストに参加してみてください。

ままシェルモードで実行して結果を読み上げてくれます。

このほか、メールはMewで、シェルモードを使えばUNIXコマンドラインが、FTPモードでは……と対応しているアプリケーションを挙げるときりがないのですが、私もほぼEmacsだけで「暮せて」いるので、視覚障害者の方が実用的に使える環境だというのはよくわかります。

Emacspeakのインストール

さて、自分のEmacsをサイバーに喋らせるにはどうしたらよいでしょうか。Emacspeakは、RPMなどがあるわけでもなく、現状ではLinux上で動くフリーまたは無料の日本語音声合成エンジンがないという問題もあります（コラム「マルチリンガルEmacspeakとBEP」参照）。ですからここでは、バイリンガル版Emacspeakをインストールするには何が必要か、という目安を述べることにします。

■動作環境

まずはサウンドを再生できる環境を構築しておきます。機種や環境によって難しさが異なるでしょうが、Linux Sound HOWTO（日本語版はhttp://www.linux.or.jp/JF/JFdocs/Sound-HOWTO.html）などを参考に、いろいろチャレンジしてみてください。なお、現在のバイリンガルEmacspeakは、Emacs 20.5とegg4の組み合わせに対応しています。

■音声合成エンジン（英語・日本語）

オリジナルのEmacspeakでは、英語の音声合成エンジンとしてIBMから無料で入手できるViaVoice Outloudが使えますが、Linux上のバイリンガル版では未対応です。

日本語の音声合成エンジンは、BEPのメンバーが開発したもので、PCM音声波形を吐くLinux版音声合成SDK（クリエートシステム開発製）というツールキットを使用しています。音声合成SDKはフリーのソフトウェアではなかったので、これまでEmacspeak公開の障害となっていましたが、近日中にベクターデザインのWebページ（http://www.vector.co.jp/）から販売が開始され、手軽に入手できるようになりそうだということでした。

オーディトリアイコンもサウンド再生ソフトを駆動するので、オーディトリアイコンと音声の両方を使いたい場合は、両者を合成するesoundというソフトウェアをインストールする必要があります。これはGNOME環境をインストールすると一緒に入る場合が多いようです。

BEPのメンバーにオーディトリアイコンだけでも利用したい人がいるのではないかと尋ねたら、何も声を出さないダミーの音声合成エンジンを用意するか、.emacsかショートカット操作でelisp変数のセッティングを変更してスピーチエンジンをオン／オフするように改造できそうだ、ということです。実現すれば、Emacs環境もサウンド効果の楽しみを享受できることになります。

■elisp

これは、elispのプリコンパイルなどをするのでなけれ

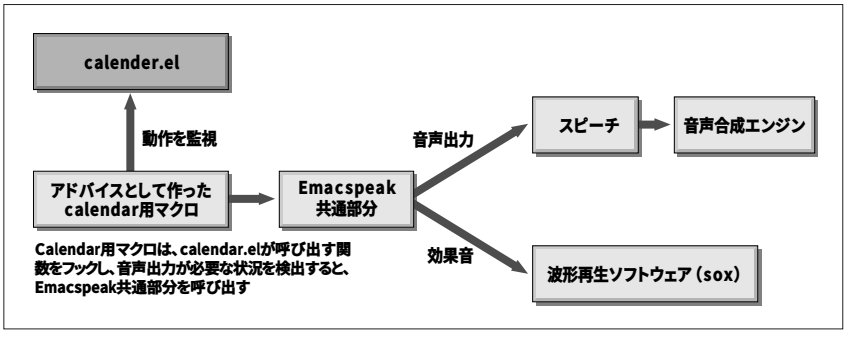
Column

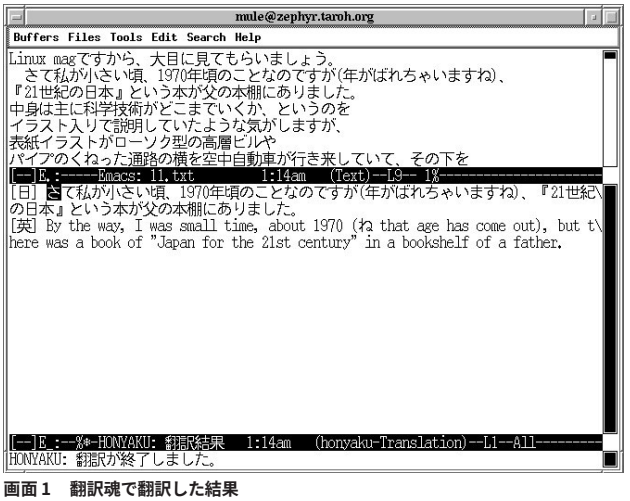
Emacspeakのしくみ

ここでちょっとEmacspeakの技術的な面をのぞいてみましょう。EmacsをEmacspeak化するには、Emacspeakのelispマクロ群を「追加」します。鋭い読者なら、「あれ？　使用しているelispアプリケーションによって読み上げ方が違うじゃん」と気づかれるでしょう。

Emacspeakのマクロは、アドバイスと呼ばれるelispの機能を使って動作しているのです。アドバイスは、特定のelisp関数が呼ばれるたびにEmacsに何かさせるように、あとから追加できるものです。elispアプリケーションごとに「必要部分だけを喋る」アドバイスを定義してやることは必要ですが、元のelispマクロを改造する必要がありません。

elispで書かれたEmacspeakのマクロは、音声合成エンジンに何を送るか決定したら、それを英語や日本語の文字列にして送り出してやります。なお、ここで「音声合成エンジン」と呼んでいるものは、ハードウェアの音声合成装置（DECtalkなど）とそれへのインターフェイスであることもあるし、ソフトウェアで実現されていることもあります。





画面1　翻訳魂で翻訳した結果



画面2　訳語の別の候補を表示させる

ば、ただ＊.elというファイルを、emacs-load-pathの通ったディレクトリに置くだけでよいでしょう。コラム「Emacspeakのしくみ」で触れているように、今までのマクロは改造なしですみ、ほかのディレクトリにあるファイルを置き換える必要はありません。

翻訳魂でゴー

日本人は英語が苦手だといわれます。英語の文を見せられて、突如急用を思い出す読者もいらっしゃるのではないでしょうか。でも、Emacs上で仕事をしているのですから、Emacsにお手伝いさせないテはありません。

「翻訳魂」とはまた冗談のようなネーミングですが、Wnn6と同じオムロンソフトウェアの商品です。翻訳魂はいかにもUNIXらしく、翻訳サーバというデーモンが動いていて、TCP/IP経由でユーザーの要求があると、翻訳作業をして結果を返してくれるのです。したがって、いろいろなインターフェイスが考えられますが、ここではEmacsから使う例を紹介します。

■インストール

翻訳魂をEmacsから使う場合、付属のelispパッケージが対応しているのはMuleバージョン2.3（Emacs 19.28/19.34）またはEmacs20.3となっています。

翻訳魂のパッケージには、多くのディストリビューションに対応した簡単インストーラとマニュアルが付いていますので、特に心配はないでしょう。対応ディストリビューションなどの情報はhttp://www.omronsoft.co.jp/SP/pcunix/taiou.htmlを参照してください。商用ソフトなので、セットアップに関してあまり詳しく述べる必要はないと思いますが、ほんとうに簡単です。

■翻訳してみよう

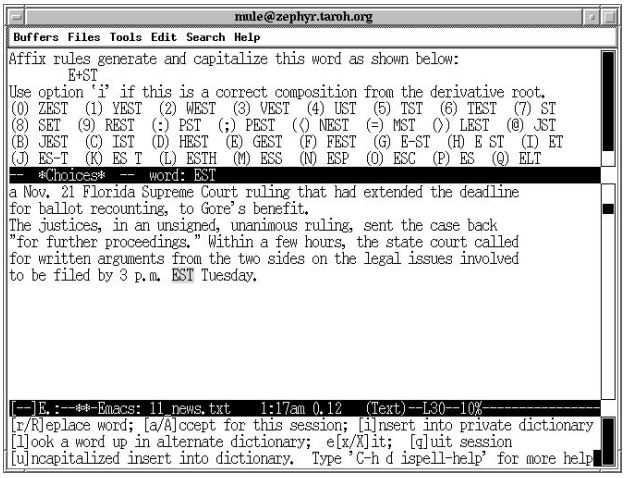
翻訳魂のEmacsインターフェイスは、まったくもってEmacs流です。実はXのGUIクライアントもあるのですが、Emacsクライアントのあまりの快適さに、筆者はそちらを使ったことがほとんどありません。Webブラウザからコピー&ペーストして……というときも、あとの再利用を考えるとやっぱり一度Emacsにペーストしてから翻訳魂のEmacsインターフェイスを使う、という具合いです。

一番よく使うのは、選んだ範囲を英訳／和訳する、**C-x y j**／**C-x y e**です。原文の言語に合わせて最後がj／eになるわけですね。これさえ覚えておけば、ちょっとした辞引きにも使えて、最高に便利です。どっちがどっちだか忘れて間違えても大丈夫。原文が表示されるだけです（日本語を和訳しても表示は日本語）。

翻訳サーバに渡す量は一度に1文のようで、マニュアルなどのように大量の文章を翻訳させても、時間はかかるけど大丈夫です。ただし、1文が極端に長い場合や、ゴミが混ざっている場合は、サーバが落ちることがあります。

翻訳が終わると、「HONYAKU: 翻訳結果」というバッファが開き、［日］［英］というヘッダの後ろに1文ずつ原文と対訳が表示されます（**画面1**）。

出来心で翻訳結果を逆に翻訳してみたくなることがありますが、翻訳結果のバッファだけは翻訳ができないので、試したければ＊scratch＊などにコピー&ペーストして翻訳してみましょう。こうして遊んでみると、和英の翻訳エンジンと英和の翻訳エンジンで得意な言い回しが違っていたり、逆翻訳の結果が原文にならないことに気付きます。開発者が聞いたら怒りそうですが、筆者の周りでは、和→



画面3　ispellを使ったスペルチェック

英→和→英→……を繰り返したりします（笑）。

■翻訳のヒント

さて真面目な話に戻して。一般的に機械翻訳のクォリティを上げる秘訣は、翻訳の前と後に文章をリライトすることだ、と言われています。筆者は英語ネイティブでないので、英訳されてできた英文の品質はよほどひどくない限りわかりませんが、和訳されてきた日本語はそのままでは読むのがちょっとつらいときもあります。それでも推測を働かせれば読めますし、すぐ上に原文が表示されているので、辞書引きの手間が省けるだけでもありがたいことです。

日本語を英訳するときのコツは、主語を省略しないことでしょう。句の意味を調べるために適当な範囲を指定して訳してみると、勝手に「私」を主語として補って、命令文にしてくれることがあります。

さて、翻訳のあとのリライトを助けるためか、「HONYAKU: 翻訳結果」バッファの原文の上で〈SPC〉を押すと、ポインタが置かれている単語の訳が複数表示されて選択できます（**画面2**）。ここで単語を選択し直すと、新たに生成された訳文が先の訳文の上に挿入されます。この手間が面倒なので、「ほかにどういう訳語があるのかな」というときくらいしかこの機能を使いません。しかし、ちゃんと訳が完成するまで真面目にこれを繰り返せば、学習して次回の翻訳に反映されるので、ほんとうは文章が完成するまで適当な語を選び続けるべきかもしれません。

また。大文字と小文字の区別は結構シビアなようで、強調のために1単語丸々大文字になっているマニュアルや、ハッカー流に全部小文字のネットニュースやメールなどは、知らない単語が含まれているものとみなされて、訳文全体の文脈がめちゃくちゃになってしまうようです。

コマンド	説明
M-x ispell-buffer	バッファをスペルチェック
M-x ispell-region	範囲をスペルチェック
M-x ispell-message	メッセージをスペルチェック（引用部分は除外）
M-$ (ispell-word)	ポイント位置の単語をスペルチェック
ispell内の操作	
?	簡易ヘルプを表示
r	指定した単語に置き換える
l	辞書の検索（ワイルドカードを使用可能）
〈SPC〉	そのときだけスペルをOKとする
a	Emacsを終了するまでそのスペルをOKとする
i	辞書にそのスペルを登録する

表1　ispellでよく使用するコマンド

ispellで綴りをチェック

ispellは、GNUのフリーソフトウェアで、コマンドラインから使うことができるスペルチェックツールですが、これまたEmacsとの相性がとってもよく、欧米人には「とても賢いスペルチェッカ」と評判です（**画面3**）。

ispellを使うのに必要なelispパッケージは、Emacsに標準で付属しているので、.emacsなどの変更も必要ありません。だから、M-x ispell-なんとかで「ispellがありません」と言われたら、ispellのインストールが必要です。ispellが最初から入っているディストリビューションもありますが、パッケージになっているものもよく見かけます。

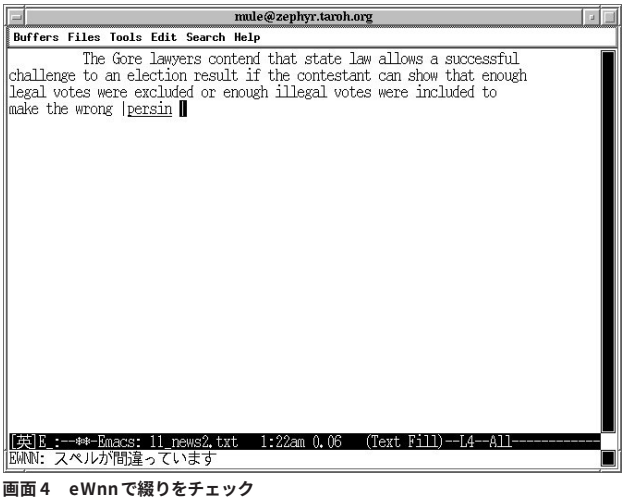
ispellは明示的に起動するものなので、ショートカットコマンド一発というものはあまりありません。使い方の詳細はinfoの［emacs］-［spelling］にありますが、よく使用する起動方法とispell内の操作を表1にまとめます。

特筆すべきは、ispell操作の**l**コマンドです。日本人が単語をカタカナ英語で覚えていると、とんでもない綴りになっていることがありますが、「これは絶対大丈夫だろう」という部分だけわかっていれば検索ができます。たとえば、“ collaboration ”を“ corrabolation ”と綴ってしまったら、いくらispellが賢くても誤り訂正候補には表示されません（間違いパターンが欧米人向けなんですね）。しかし、“ co*bo*tion ”なら候補が表示されるはずです。

eWnnで英語力向上

フリーソフトで固めなくてもいいやという場合は、ispellより高機能なオムロンソフトのeWnnがお勧めです。

「え？　英語の入力にWnn？」と思われるかもしれませんが、そもそもeWnnは、Wnn6のカタカナ英語変換機能（例：「こんぴゅーた」→「COMPUTER」）を強化し



画面4　eWnnで綴りをチェック



画面5　eWnnを和英辞典として使う

て単独の製品になったようです。動作環境については翻訳魂と同じWebページを見てください。インストールに関しては翻訳魂との共通点も多く、印刷したマニュアルも翻訳魂より充実しているので、初心者には気が楽でしょう。

eWnnも、これまたクライアント・サーバ方式をとっていて、Emacsのクライアントは製品に含まれています。クライアントはオープンソースなので、別のユーザーインターフェイスを開発する人には参考になるでしょう。

eWnnサーバは、英和・和英辞書の検索機能には独自の辞書を用いているようですが、スペルチェックにはeWnn用に拡張されたispellを利用しているようです。

eWnnの特筆すべき機能はいろいろありますが、筆者が個人的によく使用する機能を次に挙げます。

・リアルタイムスペルチェック（画面4）

Wnn4/6の起動よろしく、**C-x C-**を入力すると、左端の入力文字種の表示が［英］になります。なぬ？　フェンスモードで英文入力？　と思って入力していると、スペースなどの区切り文字を入力するたびにリアルタイムでスペルチェックをしてくれます。



画面6　eWnnで英単語の意味を調べる

・辞書引き

C-x w tでマーク位置の辞書を引いてくれます。この辞書は研究社の新英和中辞典なので、例文もしっかりしていてかなりの情報量があります。

・単語英訳・辞書引き

英文の入力中に単語を調べるには、まずローマ字で日本語を入力して〈SPC〉キーを押し、かな漢字変換をします。候補が複数ある場合は、3つに割れたウィンドウの1つ目で〈↑〉〈↓〉キーを使って選択します。次に〈→〉キーを押すと、カーソルは対応する英単語が表示されている2つ目のウィンドウに移動し、そこで〈↑〉〈↓〉キーを使って適当な英単語を選択できます（**画面5**）。このとき、3つ目のウィンドウには選択中の英単語を英和辞書で引いた結果が表示されます（**画面6**）。和英辞典を引いて英単語をピックアップし、「どれを使ったらいいんだろう」と、用例を確かめながら悩めて、実にサイバー&クイックです。

このほかにもeWnnにはたくさんの機能がありますが、ここですべてを紹介できないのが残念です。

おわりに

さて次回は、多国語環境のためのEmacsと題してお送りする予定です。では来月まで、うれしい粗面仕上げ！！（翻訳魂）、じゃなかったHappy hacking！！

番外編 Linux Conference 2000に行ってきました



2000年11月29日～12月1日に、京都の国際会館で開催されたLinux Conference 2000 Fall（日本Linux協会主催）に行ってきたので、番外編として興味のおもむくまま、独断と偏見でレポートします。このイベントでは、8つの部屋で講演と有料のセミナーが並行して開かれ、1室を使った展示会が併設されました。

Perl対Ruby！

今回のイベントは、O'Reilly主催のPerl/Ruby Conferenceと共催でした。このために来日したPerlの父・Larry Wallさんと、Rubyの父・まつもとゆきひろさんの対談「PerlとRubyの比較」は、今回の目玉のひとつでしょう。

Rubyは、Perlの影響も受けていますが、まつもとさんが「RubyユーザーでPerlより良いという人はいるが、その逆は聞かない」とけしかけたりするのでドキドキしますが、この2人は実際には仲良しです。Larry Wallさんによると、言語の設計者は設計段階でいろいろな可能性を取捨選択しているはずで、出来上がりが違う言語になっていても同じ悩みを通ってきた同志なんだそうです。

Perl6はどうなる？

さて気になるPerlの今後ですが、最終日午後のLarry Wallさんのキーノートスピーチ“Camel Lot #6/The Once and Future Perl”（らくだ6号／Perlの今までとこれから）は、ユーモアたっぷりの説明で、会場は笑いの渦に包まれました。

話の前半はPerlの生い立ちです。日ごろPerlを愛用していて「仕事のやり方はひと通りではない」（同じ仕事をするスクリプトをいろいろに書ける）というPerlの哲学は身に染みていましたが、Larry Wallさんが何を考えてPerlを進化させてきたか知ることができて、目からウロコが落ちる思いです。

Perlの目指していることのひとつは自然言語です（Larry Wallさんのルーツのひとつは自然言語処理だそうです）。同じ仕事を何通りにも書けるとか、習うより慣れろ、ある程度の曖昧性を許す、といった哲学は、なるほど自然言語と共通する部分です。

後半は、Perl6がどうなるかというより、仕様がどのように決まるかという開発体制についてが中心でした。Perl6の仕様は、Larry Wallさん個人の考えではなく、RFCとしてみんなから寄せられた提案をLarry Wallさんが「編集」して作っているということです。話の随所から感じられたことは、Perl6はPerl5までとは全然違うものになるかもしれないということです（互換性すらどうなることやら）。

最後に、スライドのために使っていたラップトップPCのデスクトップ環境を写し出し、寄せられた提案にコメントを入れ、good、bad、ugryといったディレクトリに整理するようすを見せてくれました。Larry Wallさんがどのようにperlの開発を進めているかを垣間見る、貴重な機会だったと思います。

Larry Wallさんの「次」は？

Larry Wallさんは、Perl以外にも多くのフリーソフトの作者として知られています。diff出力を食わせるとぶつぶつ喋りながらソースファイルのバージョンアップをしてくれるpatchコマンドや、環境を調べてインストール用のMakefileを自動作成してくれるconfigコマンドには、お世話になっている読者も多いでしょう。筆者にとって、Larry Wallさんはアイドルであり、心中ひそかに師匠とお慕いしていた方だったのです（笑）。

Larry Wallさんに、30秒ほど話しかける機会がありました。「Perl6以外で、次に夢中になっていることは何ですか？」Perl6以外は眼中になかったのか「うーん……」と考え込んでしまったのですが、そのあと冗談っぽく「そうだ、日本語だよ。3カ月ほど勉強しているんだ」と答えてくれました。これは同氏のセッションや懇親会でも紹介されましたが、日本語を学んでいる目的はズバリ「アニメ」です。大好きな『うる星やつら』などは翻訳されているけど、やっぱり日本語がわからないと細かいギャグのニュアンスがね、とおっしゃいます。Linux Conference 2000のあと、京の街にアニメビデオを買いに繰り出すそうで、私のLarry Wall師の像がガラガラと……（うそうそ）。

Wnn7登場

会期中、オムロンソフトウェアが日本語インプットメソッドWnn6の後継にあたる



写真　Larry Wallさんとまつもとゆきひろさん

番外編

Wnn7を発表しました。2001年2～3月に個人向けのベータ版をリリースし、5～6月をめどに正式にリリースするそうです。

Wnn7はまだ開発最終段階ということで、開発側はユーザーの声を聞きたいと盛んに言っていましたが、フリーソフトから発展したためでしょうか、こんなところにもフリーな精神が色濃く残っています。有用な意見ならWnn7に盛り込めるかもしれません。フィードバックはwnn-info@omronsoft.co.jpまで。ディストリビューションにバンドルされるようになったら、ユーザーからの声が聞こえにくくなった、と嘆いていましたよ。

さてWnn7の特色ですが、すでにどのIMEも変換効率はかなりよくなってきて五十歩百歩なので、Wnn7はとにかくキー入力を減らすことを目指したということです。デモでは、辞書登録ではなく、ふだんユーザーが入力している文から学習させて「いつもおせわになっております。へいそはかくべつなるごこうじょうをたまわり、あつくおれいもうしあげます。」をわずか3ストロークで入力する例を見せました。

また新たに、入力誤りを補正する機能が付加されるようです。おせっかいにならないかと心配してしまいましたが、そうならないように努力するからと開発者も言っているので、リリースを楽しみにしましょう。

2001年7～9月には、同じ機能で、部署や研究室などのグループ環境向けのサーバ版をリリースし、9～10月には個人向けの次のバージョンをリリースするようです。

Linuxとユニバーサルデザイン

「ユニバーサルデザイン」というのは耳慣れない言葉かもしれませんが、いろいろな立場の人を前提に設計する、という思想です。バリアフリーが「すでにあるバリアを取り去る」ことを意味するのに対し、ユニバーサルデザインは「はじめから多様性を考えて設計する」ことを指しています。

今回のイベントでは、これに関連しておもに視覚障害のある人についてのセッションが2つ設けられました。高齢化を迎える日本では、今後、身体機能、情報機器の習熟度、言語などの障害に応じたアクセシビリティが重要になってくるでしょう。

「Bilingual Emacspeakと視覚障害者のLinuxアクセシビリティ向上」（渡辺隆行さん・井上浩一さん）のセッションでは、本文でとりあげたEmacspeakが紹介されました。今回の記事は、渡辺さんや井上さんをはじめとするARGVのメンバーに多くの便宜をはかっていただきました。この場を借りてお礼申し上げます。

後半は、さらにLinuxそのものをユニバーサルデザインにするにはどうしたらよいかという内容になりました。Linuxでは、個々のソフトの設計をちゃんとすれば、いろいろな人に合わせたインターフェイスを作れる可能性があります。

Sun Microsystemsでは、GTKにいろいろなユーザーインターフェイスを接続できるような口を作り、アクセシブルにしようという構想 (http://www.sun.com/access/gnome/)があるそうです。

さまざまなユーザーインターフェイスに変換しやすいという点で「文字」に優るものはありません。Webページを持っている人であれば、グラフィックスのロゴの代わりになるべく文字を使ったり、見栄えと内容を切り放すCSS（アクセシビリティの味方！）を使って、アクセシビリティに配慮できそうです。とくに新聞社や地方自治体などには、「大多数の普通と考えられる人」以外にもアクセス可能なWeb作りを望みたいところです。

このほか、シリアルポートにtty端末を接続してLinuxがインストールできるか（視覚障害者などにとって何がウレシイのかは本文を参照）という報告もありました。インストール方法がGUIのみのディストリビューション開発者は、検討をしたほうがよいかもしれません。



写真　Perl6はどうなる？
スライドの横軸はwhipuptitude（whipping up thing quickly= 速くものができること）、縦軸はmanipulexlty（manipulate x complex = 処理性×複雑さ）で、Larry Wallさん独自の単位。シェルスクリプトは手早く仕事ができるけど複雑な仕事には向かない。C言語は時間がかかるけど複雑なことができる。sedやawkはその中間の分野を埋められたが、Perlはさらに汎用性のある分野（グラフの右上）にも伸びる。

「視覚障害者によるLinux」（南谷和範さん）では、視覚障害者がふだんどのようにコンピュータを利用しているかが報告されました。たとえば、視覚障害者が自分でLinuxマシンを管理して使用する場合、ブートの早い段階から点字ディスプレイなどに出力できなくてはいけません。気付かなかったいろいろなことがあります。

ナノカーネルを使ったマルチOSの共存

ナノカーネルとは耳慣れない言葉ですが、簡単に言えば、OSより下層をナノカーネルという機構が奪い、CPUが複数あるように見せかける技術です。この結果、1つのマシンで異なる複数のOSを同時に動作させることが可能になります。1つ目のOSをホストOS、2つ目をゲストOSと呼び、ゲストOSには多少の改造が必要ですが、ホストOSにはその必要がありません。

「異種OS共存技術ナノカーネル方式のLinuxへの適用」（日立の新井利明さん）の発表では、この特質を活かしてWindowsをホストOSにし、LinuxをゲストOSにする例が紹介されました。ソースコードが公開されているLinuxだからこそ可能なことで、まさにオープンソース万歳、です。

複数のOSを動作させるだけならVMwareなどのエミュレータもありますが、エミュレータより効率がよく、2台のPCでは絶対にできないこともあります。このセッションでは、2つのLinuxを動作させてセキュリティを高める例が紹介されました。ホストOSはメインのサーバ業務に徹し、ゲストOSにはナノカーネル経由でホストOSの動作状況を監視できるように改造が加えられています。デモでは、ホストOSが落ちたとき「落ちた」メールがゲストOSからやってきます。これに対して「Subject: reboot」というメールを返すとPCが再起動し、会場から大きなざわめきがわきあがりました。

Linux日記

第17回　メール配送（4）

リモートホストへメールを配送する場合、配送すべきホストはどのようにして決定されているのでしょうか？　今回は、MTAとDNSサーバの関係について解説します。



文：榊 正憲
Text : Masanori Sakaki

illustration ; Aki

筆者が仕事に使っている各種マシン（Windows、BSD、Linux、Macなど）は、ここしばらくプリンタが接続されていなかった。接続されていなかったというのは正確ではない。奥さんや子供が使っているWindows NT、Macには、カラーのインクジェットプリンタがつながっており、それらはネットワークに繋がっているので、当然、仕事用のマシンからも印刷できるのだが、ドライバ類をセットアップしていなかったのである。プリンタ環境をきちんとセットアップしていない理由は簡単で、紙に印刷するという作業がほとんどなかったからである。ごくたまに印刷が必要になると、仕方がないのでプリンタのセットアップができているコンピュータにファイルを持っていくという使い方をしていた。

最近、別の仕事の絡みで、PDF形式の300ページあまりのマニュアルを調べながら機器の設定を行う必要があったのだが、これにはまいった。目次ツリーがない形式であり、しかも各ページにノンブル（ページ番号）が振ってないという形式だったので、とてもではないが画面だけで読めるものではない。そこで印刷することにし、突発的にレーザープリンタを買った。

格段、気合を入れて印刷するわけではないので、一番安い3万円クラスのA4機である。キヤノンあたりが手堅いかなぁと思っていたのだが、価格と仕様の点で適当なモデルがキヤノンにはなく、ブラザーの機械を買った。12PPM、パラレル／USB両用、比較的大容量のペーパーカセット、トナーとドラムのカセットを別々に交換という点で、けっこうよいかなと思ったのである。メモリ拡張ができないとか、ネットワークアダプタを装着できないというのが難点だったが、家の用途はもっぱらドキュメントの印刷なので、特に困ることもないだろう。

しかし、考えてみると驚きである。A4で12PPMで3万円である（キヤノンの同クラス製品だと8PPMというのも、ブラザーを選んだ理由になった）。A4をランドスケープで使って、132文字×66行で印刷することを考えてみると、約800LPMの印字速度になる（正確には792LPM）。132文字×66行というのは、LP（ラインプリンタ）用紙の標準フォーマットである。15インチ幅の連続紙である。かつて（1980年代くらいまで）大型コンピュータやミニコンピ

Column

PPM、LPM、CPM

PPMはPages Per Minute、LPMはLines Per Minute、CPMはCharacters Per Minuteで、いずれも印刷速度を表す単位である。要するに1分あたりに何ページ、何行、何文字印刷できるかということだ。PPMはLBP（Laser Beam Printer）などのページプリンタ、LPMはラインプリンタ、CPMはシリアルプリンタの印字速度を表す。現在のように文字も含めてビットマップで印刷する場合は、CPM、LPMはほとんど意味がない。

Column

ランドスケープとポートレイト

用紙を縦長で使うモードをポートレイト（portrait）、横長で使うモードをランドスケープ（landscape）という。本来は、縦長とか横長といった意味はない。ポートレイトは肖像画、ランドスケープは風景画のことである。肖像画は縦長、風景画は横長にキャンバスを使うことに由来する用語なのだ。ちなみにこれは西洋絵画の場合である。水墨画の掛け軸はランドスケープでも縦長だなどと文句を言っても始まらない。

ュータに接続されていた標準的なラインプリンタは、800LPMから1200LPMという速度だったと思う。LP用紙はA4より大きかったとはいえ（おおよそB4くらい）、この速度を実現するために、1畳くらいの設置床面積、数百kgの重量で、2kVAくらいの電源を必要としていた（値段も安いもので数百万円だった）。今や35cm四方で10kg、3万円で、ほぼ同じ速度で印刷ができる（さらに画像の印刷までできる）。ここ20年くらいで、CPUはどんどん速くなり、メモリはどんどん広くなり、値段も下がってきたが、プリンタも例外ではなかったということだ。

レーザープリンタは、感光ドラム上をレーザービームでスキャンし、光の当たった所（あるいは当たらなかった所）にトナーを静電気で付着させ、それを用紙に転写するという形で印刷を行う（用紙上のトナーは、熱処理によって定着される）。したがって、文字だろうが画像だろうが自由に印刷できる。出力画像のビットマップをそのまま出力するだけだ。現在広く使われているインクジェットプリンタも同様だ。印字ヘッドが往復して印刷しているが、たいていの場合、出力されるのはビットマップイメージである。

それに対して、ラインプリンタは活字式のプリンタだった。用紙と活字の間に挟まれたインクリボンによって文字を印字していたのである。一番初期のプリンタは、電動タイプライタだった。しかし、タイプライタは一度に1文字しか印刷できないので遅かった。それなら1行分の活字印字機構を用意して、1行まとめて印字してしまえば速くなるだろうという発想で作られたのがラインプリンタである。1行分（132文字か136文字）の活字セットを持っていて、1行をドカンと印刷したのである。もちろん、普通のタイプライタのような印字機構を132桁分も並べるのは不可能だ。印字機構は1桁の幅に収めなければならない。そこで考えられたのがドラム式、ベルト式（チェーン式）の印刷機構である。用紙の前面を高速で活字が通過し、目的の活字が来たときにハンマーが用紙を裏から叩くという仕組みだ（**図1**）。ドラムが1回転すれば、必要な活字が必ず通過しているので、1行の印字はドラム1回転以内に必ず完了する。出力頻度の高い数字などは、1回転の間に2セット以上通過するようにすることで、帳票出力などはさらに高速に印刷することができるようになっていた（数字しか印刷しないなら、ドラム1回転で2行印刷できることになる）。

筆者はこの機構を知ったときに感動したものである。学生時代に、ジャンクのラインプリンタ（オフコン用の小型のもので、単相100V、1kW程度で使えた。重さも200kg以下だった）を買ったほどである。制御回路を自分で作って使おうと思ったのだが、結局、動かさずじまいだった（132桁分のドライバのための資金が足りなかった）。しばらく前まで、ドラムの一部、ハンマーブロックなども持っていたのだが、引っ越しのときに処分してしまったようだ。

あるとき、PC-9801のN88-BASICでラインプリンタのシミュレータを書いたことがある。各行を文字コード順に画面に表示するというしょうもないプログラムであったが、印字ごとにカセットインターフェイス用のリレーをカチカチいわせるようにした結果、ラインプリンタそっくりの音が出るようになり（音量は全然違うが）、一部で妙に受けた。チーチチチチという感じの音である。

Column

大型レーザープリンタ

レーザープリンタは活字式のプリンタに比べて格段に高速印刷ができる。ラインプリンタの置き換えに使われた大型LBPは、桁違いに高速である。大型コンピュータ用大型LBPを初めて見た時は感動したものである。LP用紙が、高速紙テープリーダで読み取られるような速度で流れ出てくるのである。当時、15000LPMくらいの速度だったと思う。毎分200枚以上（毎秒3枚以上）のLP用紙印刷が行われるのだ。

この速度になると、いろいろ余計な心配もしなければならない。LP用紙は折り畳まれた状態でプリンタにセットされる。高速で吐き出されるこの用紙の山を、同じように折り畳んでやらなければならないのである。あるいは、印刷ジョブごとに用紙を切り離し、スタッカに投入するといったオプションもあったようだ。こういったことを行う機械装置は、それなりに見事なものだった。

活字式ラインプリンタはもはや完全に過去のものになったようだが、名前だけはまだ残っている。lprのデフォルトプリンタ名がlpであるとか（lprという名前自体もラインプリンタ時代の名残だ）、パラレルポートにLPTという名前が残ってるとか、連続用紙は相変わらずLP用紙と呼ばれているといった程度のことだが。ほかにも、さりげなくラインプリンタ時代の習慣が意味もなく残っているものもある。プリンタのセルフテストで、各文字を文字コード順に印字し、印字位置が1行ごとに1桁ずつずれていくというパターンがあるが（全文字シフトパターンという）、あれはラインプリンタ用のテストパターンなのである。ラインプリンタの場合、あのパターンで活字の損耗、リボンの劣化、ハンマーの摩耗といったことがわかるのであるが、現在のLBPやインクジェットプリンタで出力しても何の意味もない。

さて、sendmailの話だ。前回は、SMTPによるMTA間のメール送信について説明した。今回はその続きだ。

Column

ラインプリンタプログラム

BASICで書いたラインプリンタのプログラムは、月刊アスキーのパロディ版、『年刊Ah!SKI』に掲載された。しばらく前に月刊ASCII誌でCD-ROMに復刻されたので見てみたが、ページにモザイクがかけられていた。説明を読むと、商標などの問題があるもの、今となっては意味不明なものはこのような処理云々と書いてあった。商標の問題などはまるでなかったはずだから、「今となっては意味不明なもの」ということなのだろう。

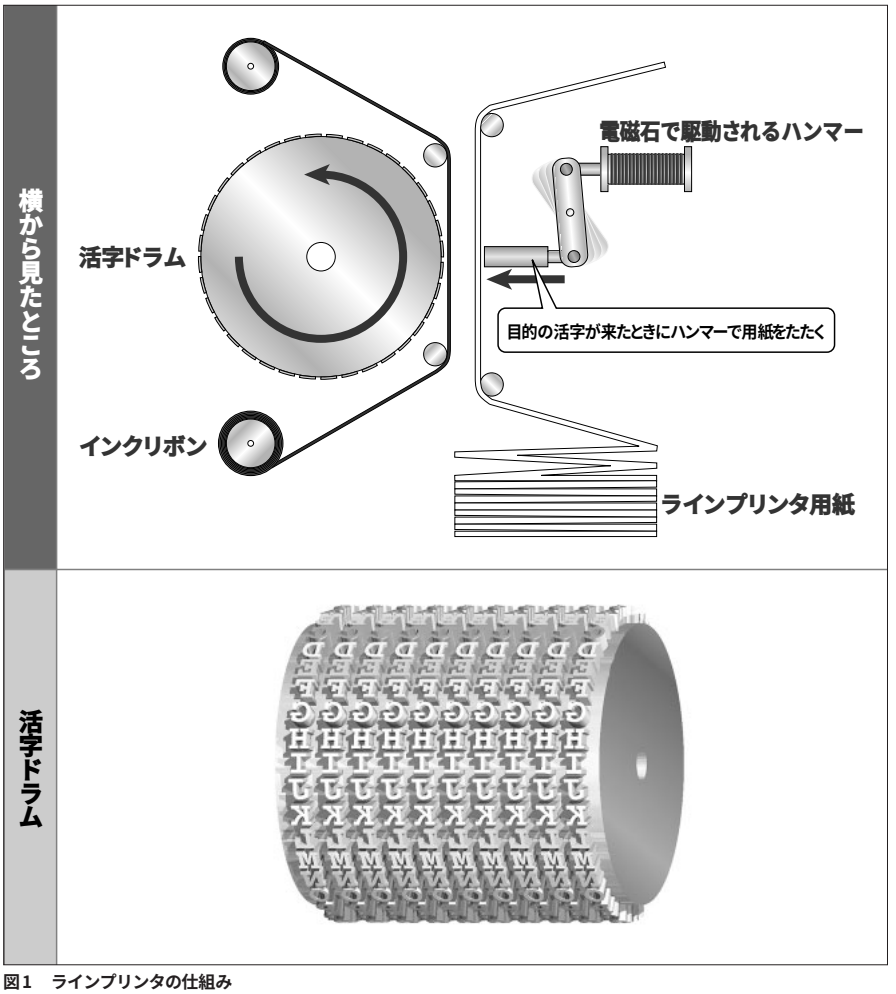


図1　ラインプリンタの仕組み

■宛て先ホストの判定

SMTP配信により、MTAはインターネット上の任意のSMTPサーバにメールを送ることができる。しかし、接続する相手はどのように判定するのか？　元になる情報が宛て先メールアドレスであることは明らかだ。“ユーザー名@ドメイン名”というアドレスのドメイン名部分から接続するSMTPサーバのホストを求めることになる。

以前にDNSの解説記事で触れたように、これにはDNSのMX（メールエクスチェンジャ）レコードで定義された情報を使う。DNSでは、任意のドメイン名に対して、メールエクスチェンジャ（メールサーバ）ホストを定義することができる。これは、あるドメインに対してメールを送りたい場合は、そのドメインに対して定義されているメールエクスチェンジャにメールを送ればよいという指示情報である。

たとえば、アスキーのネームサーバに、ascii.co.jpというドメイン名に対して、mail-svr.ascii.co.jpというメールサーバホストを示すMXレコードが定義されていれば、user@ascii.co.jpという宛て先のメールを配信するMTAは、mail-svr.ascii.co.jpに接続し、そのメールを配信する。これにより、メールサーバホスト名を含まない形式のメールアドレスが使えるようになる（**図2**）。

メールアドレスのドメイン部から、該当するメールエクスチェンジャ情報

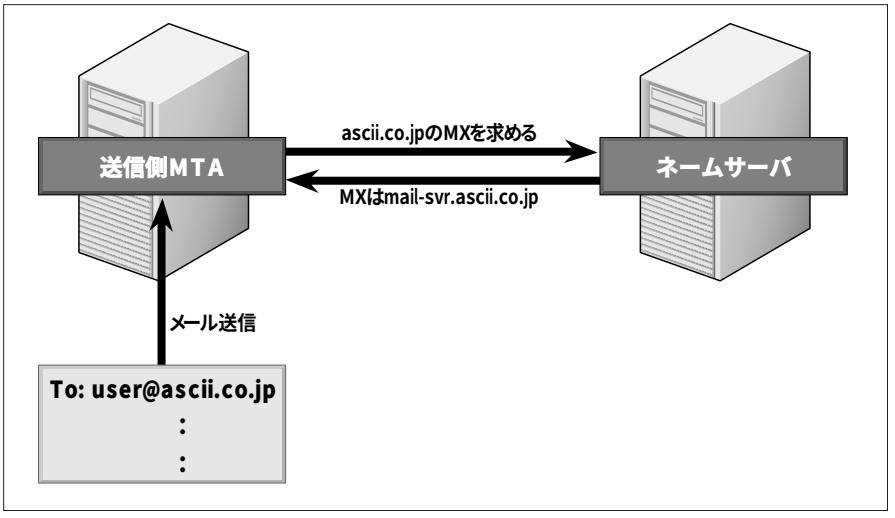


図2　MXレコードを使った宛て先サーバの判定

が得られなかった場合、2つの選択肢がある。宛て先不明エラーとして処理するという方法と、ドメイン部がメールサーバを識別するホスト名であると想定し、接続を試みるという方法だ。メールアドレスがuser@subdomain.takobeya.comという形式で、subdomain.takobeya.comにMXレコードが定義されていなかった場合、subdomain.takobeya.comというドメイン名についてIPアドレスを求め、SMTP接続を試みるのである。subdomain.takobeya.comがメールサーバホストであれば、メールを配信することができる。subdomain.takobeya.comのIPアドレスが求められなかった場合は、宛て先不明のエラーとなる。

■DNSのMXレコード

メールエクスチェンジャ情報は、組織などを識別するドメイン名（ascii.co.jpなど）に対して定義することもできるし、特定のホストを識別するドメイン名（mail-svr.ascii.co.jpなど）に対して定義することもできる。組織ドメイン名について定義すれば、その組織のユーザーに対してメールを送るときに、user@ascii.co.jpといったメールアドレスを使うことができる（**リスト1**）。

▼リスト1　ascii.co.jp用のMXレコードの定義例

```
ascii.co.jp.      IN MX    10       mail-svr.ascii.co.jp.
```

▼リスト2　メールサーバ名を含むアドレス用のMXレコード

```
mail.aru-provider.ne.jp.   IN MX    10       mail.aru-provider.ne.jp.
```

▼リスト3　サブドメイン宛のメールの処理

```
ascii.co.jp.               IN MX    10       mail-svr.ascii.co.jp.
pub.ascii.co.jp.           IN MX    10       mail-svr.ascii.co.jp.
sales.ascii.co.jp.         IN MX    10       mail-svr.ascii.co.jp.
```

▼リスト4　プロバイダのメールサーバを使う

```
ascii.co.jp.      IN MX    10       mail0.aru-provider.ne.jp.
```

個人でプロバイダと契約している場合などは、user@mail.aru-provider.ne.jpといったメールアドレスを割り当てられることが多い。この場合、mail.aru-provider.ne.jpというドメイン名のメールサーバが実際に存在するなら、MXレコードを定義していなくてもメールは配信されるだろう。しかし現実的には、mail.aru-provider.ne.jpというドメイン名に対して、mail.aru-provider.ne.jpというメールエクスチェンジャを定義するのが普通だ。メールサーバホストのドメイン名に対してメールエクスチェンジャ情報を定義するというのは、多少冗長に思えるかもしれないが、これにはわけがある。MTAは、MXレコードから得られるメールサーバにしか接続しないように設定することができるからだ。つまり、メールエクスチェンジャ情報が得られなかったときに、メールアドレスのドメイン部をメールサーバホスト名と想定した接続を試みないのである（宛て先不明エラーとなる）。このようにMTAを設定したサイトからのメールを受け取るためには、メールサーバホスト名に対するMXレコードが必要になる（**リスト2**）。

また、異なるドメイン宛のメールを1台のSMTPサーバで受信させることも可能だ。たとえば、ascii.co.jpのサブドメインを宛て先としているメールをアスキーの代表サーバで処理するとか（**リスト3**）、ascii.co.jp宛のメールをプロバイダのメールサーバで処理する（**リスト4**）といった設定を行うことができる。“ユーザー名@会社名.co.jp”というメールアドレスが使えるというサービスを多くのプロバイダが提供しているが、これはこのような仕組みを使っているのである（もちろん、メールサーバ側でもいろいろな設定が必要

だ)。

MXレコードを使うことで、メールアドレスとSMTPサーバのドメイン名を異なるものにすることができるが、メリットはそれだけではない。MXにより、メールサーバの負荷分散、バックアップが可能になるのだ。MXレコードは、1つのドメイン名に対して、優先順位(プリファレンス値)を付けて複数のメールエクスチェンジャを定義することができる。優先順位の同じメールエクスチェンジャを複数定義すれば、受信メールは各メールエクスチェンジャに分散して届くようになる(**リスト5**)。非常に多数のユーザーを管轄するメールサーバを運用する場合、このような形にすることがあるだろう。

優先順位を変えて複数のメールエクスチェンジャを定義すれば、バックアップサーバを運用することができる。プリファレンス値が小さいほうが、優先順位が高くなる。

リスト6に示した例では、mail0のほうが優先順位が高いので、通常はmail0にメールが配信される。何らかの理由でmail0が受信処理を行えない場合にのみ、mail1に配信される。これにより、mail0がダウンしているときでも、メールはmail1で処理されるので、送信側MTA上でエラーは発生しなくなる。

最近では、MXレコードは別の用途にも使われている。メールを受信する際に、発信元アドレスについてMXレコードを検索し、有効なメールサーバが存在することを確かめるという用途だ。そして有効なMXレコードがなければ、偽造メールとみなして受信を拒否するのである。ここまでの記事を読めば、発信アドレスを詐称した偽造メールが簡単に作れるということが読者にもわかるだろう。MXを検証することで、いいかげんに作った偽造メールを見抜くことができるのである(もちろん、精緻に偽造したものについてはわからないが)。

▼リスト5　同じ優先順位のメールエクスチェンジャ

```
ascii.co.jp.      IN MX    10       mail0.ascii.co.jp.
                  IN MX    10       mail1.ascii.co.jp.
                  IN MX    10       mail2.ascii.co.jp.
```

▼リスト6　優先順位の異なるメールエクスチェンジャ

```
@         IN MX    10       mail0.ascii.co.jp.
          IN MX    20       mail1.ascii.co.jp.
```

■再びエンベロープについて

前回、telnetでSMTPサーバに接続し、メールを送る実験をしてみた。SMTPセッションでは、メール本文の送信に先立って、「MAIL FROM:」で発信アドレス、「RCPT TO:」で宛て先アドレスを指定した。これがエンベロープ情報である。これとは別に、メール本文中のTo:ヘッダも宛て先を指定している。なぜこのような冗長と思われる形態になっているのだろうか?

「To: user@takobeya.com」というヘッダがメール中にあり、SMTPセッションで「RCPT TO: <user@takobeya.com>」と指定されるのであれば、確かに冗長である。エンベロープ情報などなくても、ヘッダだけ見れば十分である。しかし、メールのTo:ヘッダが複数の宛て先を指定している場合はどうだろうか?

「To: user1@takobeya.com, user2@ascii.co.jp」というように、メールのヘッダが複数のユーザーを指定している場合はどうなるのか? メール中のTo:やCc:ヘッダは、発信ユーザーが指定したままの状態で宛て先に届く。受信者は、このメールがuser1@takobeya.comとuser2@ascii.co.jp送られたことがわかる。ところで、この2つのアドレスを管轄するメールサーバは異なるものだろう。送信側のMTAは、たとえばmail.takobeya.comとmail-svr.ascii.co.jpの2台のSMTPサーバに対して別々のSMTPセッションを張り、

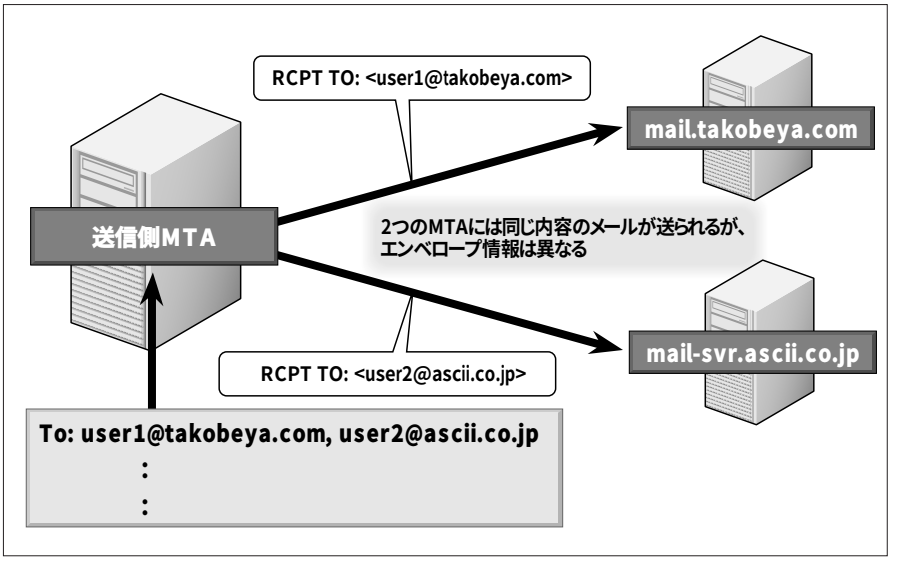


図3　To:とRCPT TO:の内容が異なる場合

同じメールを2回送信することになる。このときのエンベロープ情報は、mail.takobeya.comに送る時はRCPT TO: <user1@takobeya.com>と指定し、mail-svr.ascii.co.jpに対してはRCPT TO: <user2@ascii.co.jp>と指定するのは明らかである（**図3**）。

もし、「RCPT TO:」というエンベロープ情報がなく、To:だけで宛て先を判定するとなると、深刻な問題が発生する可能性がある。この場合、送信元からメールを受信したmail.takobeya.comは、「To: user1@takobeya.com, user2@ascii.co.jp」という宛て先を処理することになる。user1についてはmail.localを起動し、ローカル配信を行う。これは期待どおりの処理である。user2はどうなるか？　エンベロープ情報に基づいて配信する場合はuser1しか指定されないので、user2については何も行わないが、To:で判断すると、user2も処理することになるだろう。mail.takobeya.comは、ascii.co.jp宛のメールとしてmail-svr.ascii.co.jpにこのメールを送信してしまうだろう（mail-svr.ascii.co.jpには送信元から送られているにも関らずだ）。同じことが、ascii.co.jp側でも発生し、同じメールがmail-svr.ascii.co.jpからmail.takobeya.comに送られてくる。このメールも同じように処理され、ローカル配信とリモート配信を行ってしまうだろう。この処理が延々と繰り返され、1通のメールがどんどん増殖してしまうことになる。メール中のTo:ヘッダではなく、エンベロープ情報に基づいてメールを配信することにより、このような事態にならず、意図したとおりに配信することが可能になる。

MTA間でやり取りされる宛て先のエンベロープ情報は、もともとはTo:ヘッダから得られたものであるが、To:ヘッダからのエンベロープ情報の生成は、メール配信の最初の段階（MUAからMTAに渡される段階など）でのみ行われ、以後、MTAはエンベロープ情報に基づいて配信を行うことで、このような問題を解決できる。以前のmail.localの実験、telnetでsendmailと対話した実験などで、配信はエンベロープ情報に基づいて行われており、To:ヘッダは関係なかったことを思い出そう。また、Bcc:で宛て先を指定した配信は、メール中にTo:やCc:による宛て先指定がなくても配信できるが、これはBcc:に基づいて作成されたエンベロープ情報があるからだ。

■sendmailの動作

さて、しばらくメールシステム全般の話が続いたが、ここでsendmailがどのように動作するかを見てみよう。基本的に、sendmailには3つの動作モードがある。

・ユーザーのMUAプロセス中からメール送信のために実行される。

前にも説明したように、MUAからメールを送信する際に、MUAユーザープロセス中から子プロセスとして起動され、MUAから送信するメールを受け取り、それを配信する。このプロセスは配信が完了すると終了する。

・外部から配信されるメールを受信するために、デーモンとして動作する。

システムの起動時にデーモンとして起動され、外部からSMTPで配信されてくるメールを待ち受けている。メールを受信するとその宛て先などを解釈し、ローカル配信、中継処理などを行う。このsendmailはデーモンとして動作しているので、終了することはない。

・メールシステムを管理するための管理ツールとして実行される。

sendmailは、エイリアスデータベースの更新、システムやキューの状態の表示といった管理機能も備えている。このような動作モードで起動された場合、sendmailは配信処理は行わず、データベースを更新したり、メールシステムの状態を表示して終了する。このような用途に使うために、sendmailにはnewaliases、mailqといった別名のリンクが用意されている。

これらとは別にテストモードがある。このモードでは、sendmail.cfに記述された内容を検証するために、与えられたパラメータをsendmailがどのように処理するかを対話的に調べることができる。このとき、メールの配信は行わない。

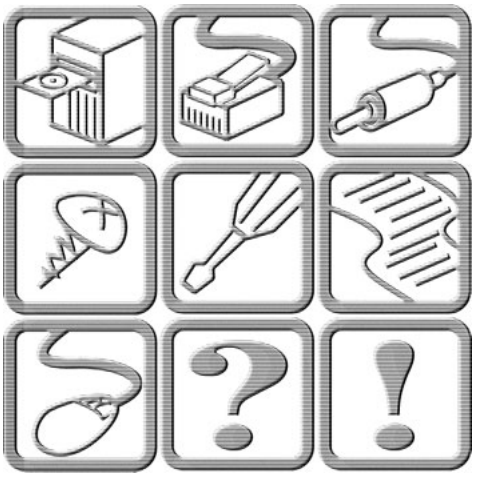
sendmailの動作モードは、コマンドラインオプション、コマンド名によって決定される。sendmailの実行時に、-bオプションで動作モードを指定する（**表1**）。

今回はここまでで誌面がつきた。次回は、sendmailの働きを動作モードごとに解説する予定だ。また、配信エージェントについても説明しよう。

オプション	動作
-bd	デーモンモードで起動する
-bD	デーモンモードで起動するが、フォアグランドで動作する。デバッグなどに使う
-bh	ホストの状態を表示する（hoststat）
-bi	エイリアスデータベースを再構築する（newaliases）
-bm	与えられたメールを送信する（デフォルトモード）
-bp	メールキューの内容を表示する（mailq）
-bt	アドレステストモードで起動する。ルールセットの評価は行うが、配信は行わない

表1　sendmailの動作モード（抜粋）

Try & [trái] Try



ボトルネック

文：政久忠由
Text : Tadayoshi Masahisa

年の瀬も押し詰まり、我が家の近辺でも（僕の活動時間でもある）夜から朝にかけて、道路で工事が行われている。いつもの風景といえば、そうなのだが、今年の工事は通例の予算消化的なものではないようだ。ふり返ること数カ月前、この街道沿いの街路樹がすべて撤去された。たぶんイチョウの木とツツジ科の常緑低木だったと思う。その時は、木々が成長し過ぎたことと、落ち葉の処理が大変だからだと思っていたが、実は違う目的があったらしい。

これらはすべて、歩道と道路にかけて、上下水道、ガス管、電気、通信ケーブルを埋設する共同溝を設置するためのものだった（よく考えてみるとそのような通知がポストに入っていたような気もしないでもない）。省庁が再編されることもあってか、やっと身近なところでもその作業が行われるようになったようだ。

これでようやく景観をもっとも害する電柱・電線がなくなり、空（どんよりとしているんだけどね）を快適に見られるようになるだろうし、通信ケーブルも新規に高性能なものが敷設し直されるに違いない。もうすぐ我が家にFTTHのような高速光通信網が…、と皮算用をしているのだけれど、実際にそれがどのようなスケジュールで実施されるのかは、まったく見えていなかったりする。そのうち近所のNTTの基地局に行って聞いてみようと思っているのだが、まぁ近々に可能だったとしても個人で使用するのに見合ったコストじゃあないだろうなぁ、今だと、きっと。もっとも、10Mビット／秒を超えるような性能を切実に必要としているわけではないんだけどね。

ボトルネック

今回は、ボトルネックについて考えてみようと思う。

コンピュータでいうところのボトルネックは、全体の円滑な進行の妨げとなるような要素・障害のことだ。一般にCPU、メモリ、ディスク、バス、シリアルバス、ネットワーク、グラフィックスなど、各デバイスとそれを連結する経路の性能のバランスが問題となるわけだが、利用者からすると、システム全体のバランスがよくても、要求に見合った応答がない場合、要するに遅いと感じたら、改善策を検討する必要があるだろう。

コンピュータを構成する各デバイスがバランスよく進化すればよいのだが、現実はそのようにはなっていない。CPUとグラフィックスだけが先走って、それ以外のデバイスは置いて行かれているのである。グラフィックスではAGPという専用バスを設けることである程度、帯域幅の問題を解決し、CPUでは2次キャッシュまでをオンチップ実装し、さらにメインメモリをRambusやDDRにすることでボトルネックを抑えようとしている。またこれら以外のデバイスも着実に性能が向上し、PCIという32ビット33MHz汎用バスの性能自体が大きな障害となってしまっているのが現状だ（今でも64ビット幅のPCIもあるが、実装されているのはサーバ用途のごく一部だけ）。広帯域

が必要なデバイス用にAGPのような専用バスを設置してもよいが、いい加減小手先の改善ではなく高性能な汎用バスが望まれている。バス幅を広げる、動作周波数を上げる、DDR（ダブルデータレート）、QDR（クワッドデータレート）技術などが取り立たされているが、どうなるか正確なところはまだわからない。ただ問題はディスクとネットワークだけなので、それらを専用バスにする、もしくは専用バス接続した汎用シリアルバスを利用することで解決できなくもない。

ともかく一般ユーザーには、将来的な問題はなるようにしかならないので、現状を見つめてみることにしよう。



CPUの性能は足りてる？

利用者にとって最も重要なことは遅いといったストレスを感じないことだが、これはそれぞれの処理内容に依存し、また気分的な傾向も加味されるので、定量化することは難しい。また遅いと感じたからといって、コストを無視するわけにもいかない。結局、性能面ではなく、懐具合とコストパフォーマンスで購入を決める利用者がほとんどであろう。

CPUの場合、というか一般ユーザーの利用するPCの場合、統計上のCPU利用率は極めて低い。多くのユーザーは、処理のリアルタイム応答性を求めていて、一日を通してCPUを有効的に活用しようと考えるユーザーはごく少数であるためだ。音楽データをエンコードするにしても、わざわざプライオリティを調整したり、実行開始時間を指定したりしてバッチ処理をスケジュールしたりはしない。

僕もPentium200MHz程度の性能の頃までは、数時間単位で時間がかかるコンパイルなどはatコマンドやbatchコマンドに登録して、寝ている時間に実行するよう心がけていたんだけど、今では処理に数時間かかるものは少ないので、その場でまわしてしまっている。

そういうわけで、長期的なCPU利用率をうんぬん言っても意味がないだろうから、短期的なものだけに注目しておこう。

短期的な処理としては、アプリケーションの起動を指示してから実際に利用可能になるまでの時間やその応答性能なども含まれるが、これらが遅いと感じたとき、それがCPUを起因とするものなのかどうかをチェックすることが大切だ。

これには、topコマンドやgtopユーティリティ、xosviewなどで、システムのCPU使用率やメモリ効率、プライオリティなどを監視するとよいだろう。またコマンドラインでは、シェルのtimeコマンド、/usr/bin/timeコマンドでチェック対処のプロセスが実際に使用したCPU時間、さらにユーザーモード時間、カーネルモード時間、CPU使用率、ページング状況などを見られる。

これらをチェックすることでCPUの性能がボトルネックかどうかを判断することができる。またシステム全体の状況をチェックするのに、procinfoコマンドなどを利用するとよいだろう。

```
$ procinfo
Linux 2.4.0-test11 #1  11 22 17:29:10 JST 2000 1CPU
Memory:   Total    Used     Free     Shared   Buffers
Cached
Mem:      126912   105576   21336    0        17640
59172
Swap:     0        0        0
Bootup: Thu Dec 14 01:26:25 2000
Load average: 0.20 0.05 0.02 1/43 2559
user :    0:04:04.61         0.3%     page in : 463697
nice :    0:00:00.10         0.0%     page out: 82906
system:   0:03:19.83         0.2%     swap in : 0
idle :    1d 1:34:32.60      99.5%    swap out: 0
uptime:   1d 1:41:57.13      context : 7239613
irq 0:    9251714  timer     irq 10:  0        via82cxxx
irq 1:    3  keyboard        irq 11:  1877393 eth0
irq 2:    0  cascade [4]     irq 12:  0 usb-uhci
irq 6:    227                irq 14:  438004   ide0
irq 7:    0  acpi            irq 15:  2        ide1
```

起動してから約1日経過した状態だ。メモリの利用量は多いように見えるが、その8割はバッファとファイルキャッシュで、システムとアプリケーションで20Mバイト程度使用しているだけだ。スワップファイルは使用していない。このシステムでちょっと特徴的なのは、ユーザー時間とカーネル時間の割合だ。カーネルモードで動作するモジュールの利用頻度が高いことがわかる。ただ、1日でCPUをこの程度しか利用していないのか、ともいえる。割り込みの回数からはシステムの利用方法も見えてくる。僕の場合は、基本的にネットワーク越しに利用しているのでeht0の割り込み回数が非常に多いが、キーボード割り込みは皆無だ。ディスクはそれなりに使用していることがわかる。

うむ、こういう情報をあらためて数字として確認すると、

常時電源を入れていることにも虚しさを感してしまうなぁ。こりゃあ、パワーマネージメントは必須だね。

メモリあってのCPU

コンピュータで最も重要な部分は、本来メモリである。現状ではCPUがクローズアップされているが、将来的には演算機能付きメモリがコンピュータの頭脳として搭載されることになると予想されている。まぁあくまで予想だ。

現状はCPU主導だ。今問題となっているのは、CPU（GPU：グラフィックス部分のも）の性能アップに本来の主役であるメモリの性能が追いついていないことだ。

■メモリのリード／ライト性能

そこで実際にメモリシステムがどの程度の性能なのかをテストしてみたいと思う。

最初は読み取り性能と書き込み性能をそれぞれチェックしたいと思い、適当な配列をメモリ上に作成し、適当な変数に代入するループとその逆を実施してみたのだけれど、読み出しは変数にdouble、書き込みはintを利用したほうが速くなってしまい、収拾がつけづらいのでmemcpyを利用したテストだけにした。

ちなみに読み出し／書き込みテストの結果は次のようなものであった（テストのメモリ配列は32Mバイト分）。

	double	int
リード：	314.5Mバイト/s	241.3Mバイト/s
ライト：	131.1Mバイト/s	188.1Mバイト/s

さてテストプログラムで利用したmemcpyはランタイムライブラリのものを使用している。ランタイムライブラリはglibc-2.2のもので、i686の設定が有効になっている。memcpyでは、あるメモリ空間から読み出し、それを別のメモリ空間へ書き込むという処理になるので、メモリバス帯域の利用効率は半分になる。また数値そのものは直接メモリ性能を表すものではなく、あくまでmemcpyでの結果である点も注意しておこう。

あと、比較対象としていくつかの環境でのテスト結果もあわせて表示したかったのだが、Linuxをインストールしているシステムは、AthlonとノートPCだけだったので、それ以外はWindows 2000上での結果となってしまった。ノートPCは両OSでテストできたのでそれを踏まえて見てほしい。

さて、これらの結果からわかることは何だろうか？

まず各メモリシステムレベルの性能比がわかる。特にIntel系に関して言えば、1次データキャッシュの半分である8Kバイト、2次キャッシュの半分である128Kバイトと64Kバイトで明らかな差となって現れている（小さなサイズのテストは誤差が大きいので注意）。2次キャッシュレベルでは1次キャッシュの半分以下、メインメモリアクセス

●Athlon700MHz／PC100CL2（Linux）	
Size:2Kバイト	1510.537510 Mバイト/s
Size:4Kバイト	1905.487805 Mバイト/s
Size:8Kバイト	1878.908129 Mバイト/s
Size:16Kバイト	792.543748 Mバイト/s
Size:32Kバイト	717.928690 Mバイト/s
Size:64Kバイト	581.298015 Mバイト/s
Size:128Kバイト	577.575293 Mバイト/s
Size:256Kバイト	528.429508 Mバイト/s
Size:512Kバイト	231.921703 Mバイト/s
Size:1024Kバイト	238.563846 Mバイト/s
Size:2048Kバイト	239.931859 Mバイト/s
Size:4096Kバイト	243.876416 Mバイト/s
●Celeron400MHz／PC100CL2（Linux）	
Size:2Kバイト	1338.673749 Mバイト/s
Size:4Kバイト	1609.497322 Mバイト/s
Size:8Kバイト	1723.852604 Mバイト/s
Size:16Kバイト	974.734872 Mバイト/s
Size:32Kバイト	992.504605 Mバイト/s
Size:64Kバイト	769.602564 Mバイト/s
Size:128Kバイト	369.055661 Mバイト/s
Size:256Kバイト	152.466915 Mバイト/s
Size:512Kバイト	145.102079 Mバイト/s
Size:1024Kバイト	144.815252 Mバイト/s
Size:2048Kバイト	144.756032 Mバイト/s
Size:4096Kバイト	144.777513 Mバイト/s
●Celeron400MHz／PC100CL2（Windows 2000）	
Size: 2Kバイト	1414Mバイト/sec
Size: 4Kバイト	1668Mバイト/sec
Size: 8Kバイト	1776Mバイト/sec
Size: 16Kバイト	772Mバイト/sec
Size: 32Kバイト	778Mバイト/sec
Size: 64Kバイト	553Mバイト/sec
Size: 128Kバイト	193Mバイト/sec
Size: 256Kバイト	155Mバイト/sec
Size: 512Kバイト	146Mバイト/sec
Size: 1024Kバイト	147Mバイト/sec
Size: 2048Kバイト	146Mバイト/sec
Size: 4096Kバイト	151Mバイト/sec
●PentiumIII 667MHz／PC133CL3（Windows 2000）	
Size: 2Kバイト	2472Mバイト/sec
Size: 4Kバイト	2761Mバイト/sec
Size: 8Kバイト	3036Mバイト/sec
Size: 16Kバイト	1336Mバイト/sec
Size: 32Kバイト	1329Mバイト/sec
Size: 64Kバイト	1339Mバイト/sec
Size: 128Kバイト	588Mバイト/sec
Size: 256Kバイト	189Mバイト/sec
Size: 512Kバイト	153Mバイト/sec
Size: 1024Kバイト	144Mバイト/sec
Size: 2048Kバイト	144Mバイト/sec
Size: 4096Kバイト	143Mバイト/sec
● Celeron 667MHz／ PC100CL2（Windows 2000）	
Size: 2Kバイト	2364Mバイト/sec
Size: 4Kバイト	2970Mバイト/sec
Size: 8Kバイト	3063Mバイト/sec
Size: 16Kバイト	1362Mバイト/sec
Size: 32Kバイト	1355Mバイト/sec
Size: 64Kバイト	551Mバイト/sec
Size: 128Kバイト	189Mバイト/sec
Size: 256Kバイト	145Mバイト/sec
Size: 512Kバイト	138Mバイト/sec
Size: 1024Kバイト	138Mバイト/sec
Size: 2048Kバイト	138Mバイト/sec
Size: 4096Kバイト	138Mバイト/sec

リスト　簡易なメモリテストの結果

となると1/10～1/20程度に低下することになる。これらのCPUの場合、キャッシュはCPUのコアと同じクロックでの動作なので、PentiumIII 667MHzとCeleron 667MHzはほぼ同じ性能であり、Celeron 400MHzとはクロック比の違いがそのまま現れている。

しかしメインメモリは、FSBとメインメモリの性能によって多少違いが出ている。FSB100MHzとPC100CL2の組み合わせのCeleron 400MHzが多少良く、次いでFSB 133MHzとPC133CL3の組み合わせのPentiumIII 667MHz、FSB66MHzとPC100CL2の組み合わせのCeleron 667MHzと続く。微々たる差だが、小さなサイズのテストと違いこのサイズのテストは、誤差が修練されてくるのでかなり実体を表している。ただしチップセットやその設定の問題もあるので、理由を断定するのは難しいが、Celeron 667MHzシステムが多少遅いのは、FSB66MHzが原因であると思われる。PentiumIII 667MHzシステムは、PC133CL3が影響している可能性があるものの、PC100CL2と取り替えてテストしてみないことには断言はできない。でも個人的には、PC133CL3はPC100CL2よりも効率が悪いと考えている。FSBが133MHzのために価格も安かったので思わず購入してしまったが、失敗だったかもしれない。PC133CL2が安くなったら、と思っていたんだけど、DDRやRambusのこともあるし、そもそもそれほど気にすることでもないんだけどね、僕の使い方だと。

で、まぁ、問題はAthlonの結果の見方である。1次データキャッシュ64Kバイト、2次キャッシュ256KバイトのAthlonの場合、32Kバイトと128Kバイトでギャップが来そうなものだが、そうはなっていない。またmemcpy性能もIntel系とは明らかに違う。キャッシュ性能はそこそこだが、メインメモリ性能は高いのである。なお、これらの結果には、テストサイズで表された数字のループ処理であることも付け加えておかねばなるまい。1Kバイトから128Kバイトまでは1000回のループ、それ以降は20回のループでその経過時間をもとに性能として表している。

これらの理由は、キャッシュメモリ自体の性能の違いもあるかもしれないが、たぶんキャッシュ利用のアルゴリズムの違いが影響していると思われる。つまり、ある意味巨大なキャッシュをコントロールするにあたって、あるコンテキストの要求でそのすべてを利用されることを優先しない設定であり、さらにメインメモリのレーテンシーを抑えることを優先してライトバックのような手法を使用していないことが考えられる。

そのため、8Kバイトまでの2Gバイト／秒、それ以降徐々に低下し、32Kバイト、256Kバイトでちょっとしたギャップが生じ、PC100CL2のメインメモリへのアクセスが240Mバイト／秒となっているのだろう。もちろん、FSBが200MHzで、クロックフォワード同期という手法も、メインメモリへ効率よくアクセスするためのよい影響となっていると思われる。

逆にIntel系は、ライトバックを利用することで、キャッシュ内で完結するような処理の効率は高まっているが、メインメモリへのアクセス性能は逆に低下してしまっている。

どちらの手法がより効率的かというのは極めて難しい問題で、つまるところ利用するアプリケーション次第としかいえない。特に今後もCPUはクロックを上げ続ける一方で、メインメモリの性能向上はこの2倍か4倍程度となると本当に厄介だ。ちなみにキャッシュのライトバックはソフトウェア的にもコントロール可能なので各アプリケーションレベルでそれを調整することが多くなると思う。なおMTRR（メモリタイプレンジレジスタ）は、キャッシュ自体ではないが、メインメモリの範囲を指定して、その範囲をキャッシュするか、しないといったコントロールを行うことができる。この機能は、PentiumIIIにもAthlonにもあり、メインメモリへのアクセスのレーテンシーが重要な場合などをはじめ、いくつかの用途では実際に利用されている（XFree86のXサーバなど）。

これらの結果から明らかなように、メインメモリアクセスは大きなボトルネックとして横たわっているわけだ。メインメモリへのアクセス効率を優先しているAthlon 700MHzでも8倍、PentiumIII667MHzにいたっては、20倍もの性能ギャップとして表面化している。これらのCPUは現状入手できる最低クラスの周波数であり、最高は1GHzを超えPentium4には1.5GHz版もある。もしこのままのメインメモリシステムを実装したのであれば、ギャップはさらに2倍以上になってしまう。そこでDDRやRambusというわけだ。あとインターリーブという手もあるが、何にしてもちょっとこのギャップは埋まりそうにない。いっそのことオンチップメインメモリシステムにしてしまったほうがいいかもね。128Mバイト程度ならそろそろ入るでしょ。

とにかく、システムを効率よく稼動させるための実装メモリのサイズなら増設すれば解決するが、メモリ性能となると利用者の選択肢は少ない。PC100からPC133への変更は、まったくの無駄ではないが、焼け石に水に近いと思う。でも次世代メモリシステムが手ごろな値段になるには

しばらく時間がかかりそうだ。まぁCPUの待ちぼうけが増えたとしても、こんなものと割り切るしかない。一応BIOS設定をいじくると多少挙動が変わると思うのでヒマならやってみるとよいだろう。

それにしてもAthlonのメモリシステム／メインメモリアクセスのアプローチは面白いねぇ。実際性能もいいし。そのうちPentium4も調べてみよっと。



ディスクサブシステム

ディスクサブシステムに関しても、利用方法によってその要求する性能は大きく違う。

Linuxで数10Mから数Gバイトの画像や動画データを扱いたいユーザーは、ストライピングを行って、PCIバスの限界の100Mバイト／秒の性能が欲しいだろうし、実際、手頃な価格のIDEドライブ3～4台でこのレベルを達成することができるので、実施している人も多いと思う。ただ100Mバイト／秒といっても、先ほどのメモリテストの結果はmemcpyのものなのでメモリの性能と比べると1／4程度でしかない。ディスク自体の性能からすると8台並列にすることで300M～400Mバイト／秒を達成できるのだが、いかんせん理論値132Mバイト／秒が32ビットPCIバスの制約なのでどうすることもできない。PCIバスは共有バスなので現状のシステムでは、別々のスロットに分散したとしても意味がない（あっそうそう、チップセットからひとつの共有PCIバスを提供するのではなく、複数のチャネルを提供すれば見かけ上は何とかなるなぁ）。まぁ64ビットPCIもあるが、しばらくは、この制約は変わりそうにない。ん～、困ったね。

で、性能が必要なユーザーがいる一方で、それほど性能が必要ない人もいる。僕もそのひとりで、統計を取ってみて必要なさそうなのでLinuxシステムのディスクは、旧世代のIBM-DPTA-352250に変えちゃいました。

リモートから使用し、利用するアプリケーションも少なく、このシステム上に主要なファイルを置いていないので、冒頭のprocinfoコマンドの結果を見てもわかるように、必要なものは完全にファイルキャッシュに常駐した状態で、バッファキャッシュも比較的多く利用でき、ディスクアクセスがあまり気にならないのがその大きな理由だ。

一応、ちょっとしたディスクの性能テストを紹介しておこう。シーケンシャルリードのテストは次のように行える。以下は/dev/hda1の先頭から256Mバイト分読み出している。

```
#  /usr/bin/time  dd  if=/dev/hda1  of=/dev/null
bs=65536 count=4096
読み込んだブロック数は 4096+0
書き込んだブロック数は 4096+0
0.01user 1.43system 0:14.23elapsed 10%CPU
```

256Mバイトの読み出しに14.23秒かかっているので、性能としては約18Mバイト／秒ということになる。今となってはちっとも速くないが、UDMA66対応でシステムへの負荷も小さいので、これを採用したわけだ。いくら速くなくてもいいからといって、PIOモードのものは決して使いたいとは思わないけど。

実際には、システムキャッシュを含め、ファイルシステム経由で利用することになるので、そのテストもやっておくことにしよう。このシステムのパーティション構成はシンプルに次のようになっていて、hda1はext2、hda2はReiserfsでフォーマットして利用している。

```
# cat /proc/partitions
major minor  #blocks  name
   3     0   21992544 hda
   3     1    4016218 hda1
   3     2   17968702 hda2
```

ではファイルシステム違いを含めてテストしてみることにしよう。それぞれのファイルシステム上で256Mバイト分のファイルの作成と読み出しの時間を計測している。

```
# sync
# /usr/bin/time dd if=/dev/zero of=ext2.dat bs=65536
count=4096
0.01user 1.68system 0:13.28elapsed 12%CPU
# sync
# /usr/bin/time dd if=/dev/zero of=reiserfs.dat
bs=65536 count=4096
0.03user 3.86system 0:14.39elapsed 27%CPU
# sync
# /usr/bin/time dd if=ext2.dat of=/dev/null bs=65536
count=4096
0.02user 1.33system 0:14.12elapsed 9%CPU
# sync
# /usr/bin/time dd if=reiserfs.dat of=/dev/null
bs=65536 count=4096
0.02user 1.61system 0:16.63elapsed 9%CPU
```

この場合、それぞれのパーティションのディスク上の位置による性能の違いやバッファキャッシュ効果の問題もあり、単純にかかった時間を比較するのは間違いで、その時間を含め、システム（カーネル）時間とCPU使用率を考慮する必要がある。これらを加味すると次のようなことがわかる。まずReiserfsでは書き込みに際して、2倍以上のシステム時間を必要とし、CPU使用率も高い。読み出しでは、Reiserfsのシステム時間の消費はext2より多少高い程度に抑えられている。つまり、知ってのとおり、ジャーナルログファイルシステムであるReiserfsは、その構造ゆえファイル生成時のオーバーヘッドが大きいのである。先のシステム時間の差は、このファイルシステムドライバの処理時間の違いを表しているのだ。

全体の経過時間としては、ファイルシステムのオーバーヘッドをバッファキャッシュが相殺する形となっている。ちなみに遅延キャッシュであるバッファキャッシュは、これらの処理の完了からデフォルトで30秒後、ディスクに書き出すようになっているのでそれも確認してみよう。

書き込むファイルのサイズがバッファキャッシュでまかなえるような場合は、次のような結果となる。

```
# /usr/bin/time dd if=/dev/zero of=ext2.dat bs=65536
count=1024
0.01user 0.42system 0:00.68elapsed 63%CPU
# /usr/bin/time dd if=/dev/zero of= reiserfs.dat
bs=65536 count=1024
0.02user 0.98system 0:01.00elapsed 99%CPU
```

基本的にメタファイルの処理は遅延処理されないため、このシステムでは、ext2だと145Mバイト／秒程度、Reiserfsだと64Mバイト／秒程度で頭打ちとなるが、ほとんどのアプリケーションがこの恩恵を授かることができる。

これを見ると僕のような使い方では、それなりのディスク性能でもいいじゃん、という理由が納得できると思う。

最後にランダムアクセスとは言い難いけど、それっぽいテストもしてみよう。これらのファイルシステムでの複数ファイル処理性能を見ておく。

```
# /usr/bin/time cp linux-2.4.0-test10.tar.gz
/dev/null
0.01user 0.08system 0:00.08elapsed 102%CPU
# sync
# /usr/bin/time tar xfz /usr/src/linux-2.4.0-
test10.tar.gz
3.32user 1.37system 0:12.35elapsed 37%CPU
# /usr/bin/time tar xfz linux-2.4.0-test10.tar.gz
3.20user 9.81system 0:16.25elapsed 80%CPU
```

reiserfsのオーバーヘッドは、ext2と比べるとかなり高い。昔Linuxを稼動させていたPentium 200MHzシステムだと、敬遠したくなるレベルだが、今時のシステムなら重荷になることはないし、むしろ何もしない時間が多くなるよりは、負荷が増加しても利用者が便利になるほうを選択すべきだろう。ただ、Reiserfsを利用することで便利になるかどうかを検討する必要はあるけれども。僕はディスクチェックが短くて済むという理由だけで選択した。

Reiserfsの信頼性はというと、ここ数カ月使用していて、問題が生じたのは1回だけ。

vs-13048: reiserfs_iget: bad_inode. Stat data of (301351 301460) not found

あるディレクトリが吹っ飛びそうになったが、ディスクチェックで回復できた。この問題が起きたのは、Reiserfsのドライバはそのままにカーネルのパッチレベルをいくつか強引に上げた時なので、ファイルシステム自体の問題というよりは、僕の運用上の問題である。結構普通に使えているのだけれど、一般にはメリットは少ないかもね。



伝家の宝刀、RAMディスク



あっそうそう、コンパイル時間を縮めるには、ディスクサブシステム全体の性能を上げてもいいんだけど、やっぱ/tmpをRAMディスクにするのが一番、というかそういう場合もある。PC100CL2の128Mバイトなら6000円ちょっとだし、/tmpには64Mバイト（16M程度でもいいかな？　4Mバイトでもある程度問題なかった）も割り当てればほとんどの場合は十分だと思う。でも一般には通常のメインメモリとしてファイルキャッシュなどとして利用したほうがもちろん効果的なんだけど、128Mバイトで十分なシステムに256Mバイトも搭載しちゃうとメモリもヒマしてるでしょ。ロト6が当たったら、何Gバイトも搭載してワークディレクトリにマウントして使おうと思っているんだけど、あれRAMディスクは256Mバイトまでだっけ？

やり方はrdモジュールをロードして、/dev/ramdiskをフォーマット、あとは/tmpにマウントするだけでいい。詳細はまた今度。

Linux ガベコレ

Linux Garbage Collection

目からウロコの用語辞典

文: しのはらひろあき
Text : Hiroaki Shinohara

第10回
【フリーソフトウェア】（ふりー・そふとうぇあ）
【オープンソースソフトウェア】（おーぷん・そーす・そふとうぇあ）
【オープンソースの定義】（おーぷん・そーす・の・ていぎ）
【ライセンス】（らいせんす）
【GNU General Public License】（ぎゅうにゅー・ぜねらる・ぱぶりっく・らいせんす）

フリーソフトウェア
【ふりー・そふとうぇあ】

（1）タダ（free）で手に入るソフトウェア。インターネットでダウンロードできるものに加え、最近ではCD-Rに何本も詰め合わせで配布される形態も増えてきた。流通場所は主に秋葉原の路上やオフ会場など。まだLinux対応のものが少ないのが残念。これらのソフトは、手に入れて使用しているとモレなく目つきの悪いトレンチコートを着た男性とお友達になれる機会が得られるほか、3食強制労働つきの無料別荘プレゼントの特典もあるなど、まさに至れりつくせり。人生を棒に振りたい人は使用を検討してもよいかもしれない。


3食個室付き

（2）自由（free）なソフトウェア。古来より「フリーソフトウェアは何から自由なのか」という疑問は、学会の議論の中心であった。今のところ、研究者の大多数を納得させられる回答は見つかっていない。経済的コストからの自由、厳しいライセンスからの自由、妻あるいは夫、はたまた編集者による束縛、締切からの自由など、多くの人々が自分勝手な夢を重ね合わせた主張を繰り返してきたが、「少なくともバグ、不具合から自由（trouble-free）という意味ではありえない」という点でのみ意見の一致が得られている。

余談だが、Windows界では、市場占有率が何ごとにも優先されるため、“シェアウェア”の台頭が著しく、フリーソフトウェアの立場は年々弱まっている。

オープンソースソフトウェア
【おーぷん・そーす・そふとうぇあ】

コンパイルするのが面倒なのでソースファイルを公開して、その先の作業は利用者にやってもらおうというDIY主義のソフトウェア。うまく動かなくても「設定がおかしいのでは？」、『“makeできません”だけでは誰もアドバイスしようがありません。最低、OS環境の情報くらいつけておくのが質問するときのマナーではありませんか？　私たちは魔法使いではないのですから、あなたのコンピュータ環境はあなた自身にしかわからないのですよ」、「ということにしたいのですね？」などとユーザーのせいにできるので、従来よりも効率的にバグの存在を隠蔽することが可能。この効果には大手ソフトメーカーも注目しており、自社プロダクトへオープンソースの仕組みを取り入れる事例が急速に増えつつある。この動きは他業界にも広がっており、「オープンソースの導入によりリコールをまぬがれた自動車メーカー」、「オープンソースで加工乳の品質向上を達成した食品企業」、「オタクソースでうまみを増したお好み焼き屋」など、各界からぞくぞくと喜びの声が届いている。

オープンソースの定義
【おーぷん・そーす・の・ていぎ】

なんでも定義づけをしなければ気が済まない、悪魔のようなLinuxユーザーは、オープンソースソフトウェアとは何かを定義する指標まで提案しだした。これが「Open Source Definition」（OSD）である。OSDによると、オープンソース

ソフトウェアはそのライセンス条項などが下記のような条件をすべて満たしていなければならない。以下、各規定はOSDより引用のうえ、編者が日本語訳してわかりやすい解説をつけておく。

1. Flea Redistribution

蚤の市で販売してもよいこと。コミケ、ハムフェアは不可

2. Source Code

ソフトウェアを配布するなら、最低限、そのソースコードが存在すること。ありもしないソフトを、さも実在するかのように騙って配布してはいけない。バイナリの実行ファイルが存在するかどうかなんてどうでもイイ。大事なのはソースだ。

3. Derived Works

2次利用してさまざまなものを作ってよいこと。オレのコードを勝手に……などと歯がみしてはいけない。どうせデキの悪いソフトを土台にして、より良いものを作ろうとしても後悔するだけなのだから、放っておけばよい。それをいちばんよく知っているのは開発者であるあなたのハズである

4. Integrity of The Author's Source Code

原著作物を制作したプログラマーを尊重すること。どんなに汚いコードでも、「ひどいコーディングスタイルですね、小学校からやりなおしたら（藁」などというメールを送りつけてはいけない。一晩中家にこもってプログラミングしているようなただでさえアブナイ人物を、完全な引きこもり（ヒッキー）に直行させる危険性がある

5. No Discrimination Against Persons or Groups

いわれのない理由で差別をしてはいけないこと。特に、デブやデブの集団にオープンソースソフトウェアの使用を禁じてはならない。なぜならオープンソースソフトウェアユーザーにはふくよかな人々が多いからである。編者も思わず感涙の規定である（編注：できれば、チビとデベソもお願いします）

6. License Must Not Contaminate Other Software

これはかなり難解な規定だが、「contaminate＝汚す」ということばに注目しつつ、編者のブリリアントな脳髄によって翻訳したところによると、雑誌の付録CD-ROMなどに収録して配布されるとき、同梱のほかのソフトの純潔を汚すようなマネをしてはならない、といった意味である。おそらく付録CD-ROMの中では、収録されたソフトパッケージ同士で「へっへっへ。いいじゃねーか、一度くらいよー」「いやっ！　あなたのようなコードに犯されるくらいなら、舌を噛んで死にます！」というような丁々発止のやりとりが交わされているのであろう。本誌付録CD-ROMも、ときおり着衣に乱れがあったり顔を赤らめたりといった状態で店頭に並んでいることがあるが、そんな事情が……と思うとインモラルな妄想の止まらない、若い編者であった

ライセンス

【らいせんす】



前もこういう絵描いた気がする……

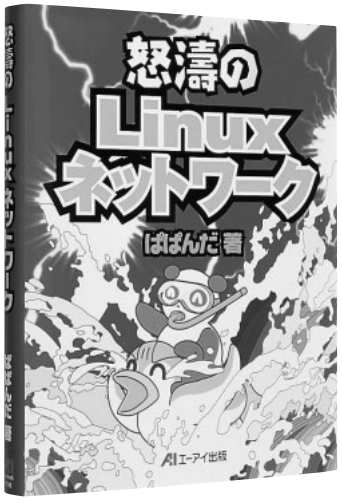
“しばり”のこと。しばりのメッカといえば歌舞伎町・新大久保の風俗産業店と思われがちだが、過度の自由に飽きた若者たちを中心に、いまライトSMが静かなブーム。これからのしばりを担うのは、彼らだともいわれている。本来、ユーザーに自由に使用してもらうことを目的として公開されているはずのフリーソフトウェアやオープンソースソフトウェアも、被加虐志向の人々によってさまざまなしばりが課されていることが多い。これは、そもそもソフトウェアのコーディング作業というものが苦行にほかならず、プログラマーは先天的にマゾヒストの素質をもっていることに起因する（一部のLinuxプログラマーの仕事場を深夜に盗聴すると、「ああっ、女王さまっ！　coreをくださいませっ！」「うるさいっ！　この三流プログラマーが！　おまえみたいなブタはキーボードをお舐めっ！」といったやりとりが聞かれるという）。

GNU General Public License

【ぎゅうにゅー・ぜねらる・ぱぶりっく・らいせんす】

略してGPL。健康のため、プログラマーはみんなで牛乳を飲もう！　というライセンス。しかし、提唱した教祖の髪の毛が伸びすぎていたせいか、肝心の牛乳の製造過程で細菌が混入（編注：これもcontamination、略してコンタミ）。この細菌が、GPLにしたがって配布されるプログラム。そしてそれを利用するプログラムにも孫、曾孫の代にわたるまで連鎖感染をくりかえすという事態を引き起こし、問題となっている。感染したプログラムはすべてGPLに従うという症状を呈するため、ある意味で最強のコンピュータウイルスともいわれる。この汚染の反省にたって、ホモジェナイズ加工で下痢を引き起こしにくくした「LGPL」も市販されている。

Books



怒涛のLinuxネットワーク

ぱぱんだ 著
エーアイ出版
B5変形判／408ページ　　本体価格 2800円

前著『Linux 怒涛のQ＆A』が大好評だった、Linux「怒涛」シリーズの第2弾。筆者のぱぱんだ氏については、ダラズ（Kondara Projectのメンバーをこう呼ぶ）としてご存じの読者もかなりいるのではないだろうか？　なお、氏は自身のホームページにおいてもLinuxの伝道師（本人談）として日夜、布教を行っておられる。

今回はタイトルからもわかるとおりLinuxのネットワークまわりの話題が中心になっている。内容的には、ハードディスクのパーティション設定のコツといったインストールのポイントについて解説するなど、初級者を意識した構成。主題であるネットワークについても、TCP/IPの基礎から、Linuxにおける基礎設定の手順、各種サーバの構築、セキュリティについての基本的な考え方にいたるまで、幅広いトピックがわかりやすく解説されている。いまが旬のフレッツ・ISDNとCATV/ADSLも取り上げられている欲張りな1冊だ。

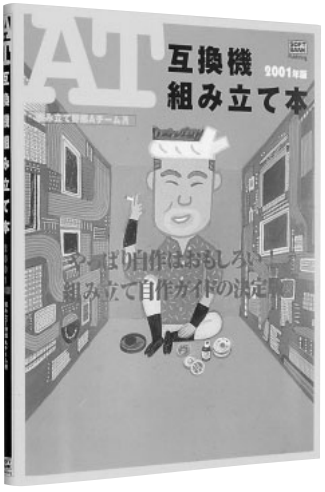
Linux 数値計算ツール　Tools for Numerical Computation on Linux

大石進一 著
コロナ社（http://www.coronasha.co.jp/）
A5判／180ページ　　本体価格 2200円

現在のパーソナルコンピュータは、ネットワーク、グラフィックス、ワードプロセッシングなど、広範な分野で利用できる汎用ツールとして活躍しているが、本来コンピュータは数値計算のために開発された機械である。

本書は、コンピュータによる数値計算の初級者を対象に、フリーの数値計算ソフトウェア「Octave」と「Scilab」の使い方を解説するものである。これらのソフトウェアは、Cなどのプログラミング言語に比べ、はるかに修得しやすいうえに、素人が作成するプログラムよりも格段に高速で安定している。前半部では、連立1次方程式、行列の固有値、数値積分、常微分方程式などの解法を示し、後半部では得られた数値解に含まれる誤差評価の方法などを解説している。仕事や研究に数値計算を必要とするエンジニアや研究者はもちろん、理工系の大学生／大学院生にも推薦したい良書だ。



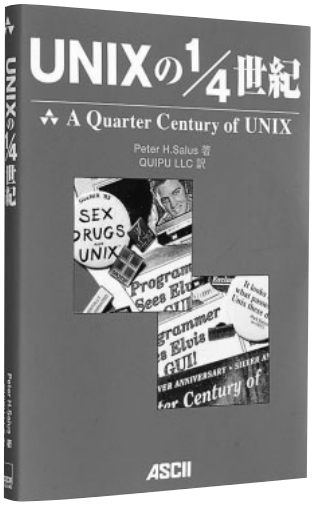


AT互換機組み立て本　2001年版

組み立て野郎Aチーム 著
ソフトバンク パブリッシング
B5変形判／186ページ　　本体価格 1800円

いまや、IBM、NEC、Dellといったメーカーが Linuxをプリインストールしたサーバーマシンを次々に発表する時代である。さらには、ショップブランドなどを中心に個人ユーザー向けのLinuxプリインストールマシンも増えつつある。しかし、手ごろな価格帯の大手メーカー製PCでLinuxがサポートされているものは、まだまだ少ない。加えて言うなら、Linuxは、やはり自らの手でインストールして、あれこれと設定を楽しみたいものだ。「Linuxには自作PCがよく似合う」というのは、わたしの思い込みだろうか？

それはさておき、パーツを選んで購入し、自宅に持ち帰ってコツコツ組み立てる作業には独特のおもしろさがある。無理に表現するなら「ばら売りのプラモデル」が近いセンだろうか。こうした感覚は基本的には男の世界である。男性読者諸君、最新ハードウェアもしっかり押さえた本書を読んで学習し、いざ行かん自作PC道を！（謝＞女性読者様）。



UNIXの1/4世紀

Peter H.Salus 著／QUIPU LLC 訳
アスキー
A5判／280ページ　　本体価格 2400円

UNIXの生い立ちは非常に複雑だ。本書は、開発に携わった多くの人物への取材を通じて、その成長と発展の経緯を明らかにするために書かれたものである。世界史になぞらえれば中世から近代にあたる期間を取り上げており、その頃の日本はまだ古代といってもよい時期であった。10年以上コンピュータとつき合いのあるかた以外にはわかりにくい部分があるかもしれない。LinuxがUNIXというOSの眷属であることは間違いない。しかし、LinuxはUNIXの子孫ではない。そのため本書にはLinuxがほとんど登場しない。あくまでUNIXの歴史書であって、Linuxの解説書ではないのだ。このように、本誌の読者にとって読みやすい本とはいえないが、Linuxのモデルであり、DOSやWindowsにまで多大な影響を及ぼしたOSであるUNIXがどのようにして作られ、どのように世界中に広まっていったのかを知ることは、あなたのコンピューティングライフをより豊かにしてくれるだろう。

UNIX Programmer's Library GNU C++プログラミング



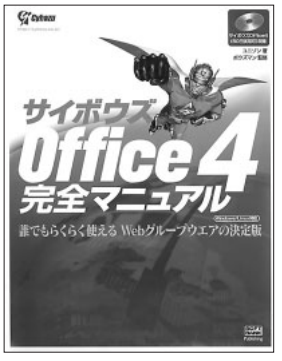
トム・スワン 著
日向俊二 監訳

翔泳社

B5変形判／992ページ／CD-ROM1枚付き

本体価格 6800円

サイボウズOffice4完全マニュアル



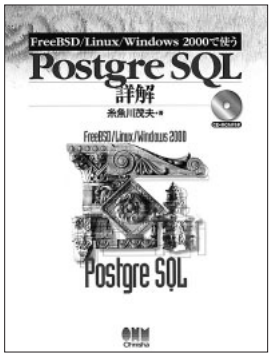
ユニゾン 著
ボウズマン 監修

ソフトバンク パブリッシング

B5変形判／268ページ／CD-ROM1枚付き

本体価格 3600円

FreeBSD／Linux／Windows 2000で使う Postgre SQL詳解



糸魚川 茂夫 著

オーム社

B5変形判／428ページ／CD-ROM1枚付き

本体価格 4500円

Red Hat Linux 7 サーバー構築入門



高原利之 著

ソーテック社

B5変形判／288ページ／CD-ROM1枚付き

本体価格 3500円

読者の声

俺にもいわせろ！

道路は相変わらず渋滞してますし、エアカーも走っていないけど、あっさりと21世紀になってしまいました。今世紀もLinuxでガンガン行きましょう。

1月号特集1へのご意見

「XFree86セットアップ徹底ガイド」は実務に役立ち、社員研修にも役立ちました。今まで私自身も理解できていなかったXFree86の謎が少し解明できたような気がします。

ところで、なにゆえにファンタジックなキャラクターを採用したんでしょうか？ それが妙に気になりました(笑)。

（伊藤真和さん）

Tux RacerをやるためにXFree86をいじっているところだったのでタイムリーな記事です。なんとか参考にしてTux Racerをプレイするぞ！

（笠井 縁さん）

毎月店頭に並んだときに立ち読み（すみません）してから購入するかどうか決めています。Linuxは「お遊び」みたいなものなので、購入する／しないはそのときの気分ですが、ディストリビューションのCDが添付されているときや、自分が持っている／購入を予定しているハードウェアの記事が載っている場合は購入して家で読ませてもらっています。Linuxは少し前のハードを購入するのが安全といった声もあるようですが、1月号のGeForce2 MXカードのような特集は大変参考になります。今後も同種の記事を掲載していただけるよう期待しています。

（柏木正元さん）

XFree86はメジャーバージョンアップをしてより強力になりました。従来のバージョン3.3系統に比べると、まだちょっと不安定のようですが、徐々に安定していくでしょう。最新のグラフィックスチップへの対応も早くなりましたね。昔はメーカーがハードウェアの情報を開示しないため、対応に時間がかかることもしばしばでした。中には、「どこどこの製品では動作しないし、メーカーが対応を改めない限り、今後対応する気もない」とドキュメントに明記されているものすらありました。

それはそうと、「謎解き」とくればファンタジックなキャラです（^^）＞伊藤さん

フレッツ・ISDNで常時接続インターネットサーバ！

ぼちぼちフレッツ・ISDN＋Linuxによる常時接続サーバが増えてきました。当方も昨年10月にLinux常時接続サーバとLinux情報サイトを立ち上げ、通信料を気にせずインターネットを楽しんでいます。皆さんもぜひトライしてみてください。

（おいでんさん）

フレッツ・ISDNに加入しようかなと考えていたので1月号の連載特集は興味深く読みました。

1月号からハガキからWebに代わりましたがどっちが便利なのでしょうね。Webになると接続しているあいだに書くことを考えなければならないので、結構難しいなぁ

（小野文夫さん）

春から就職で東京に行くことが決まっているけど、常時接続の手段として何を選択すべきか迷っている。

（ヒロさん）

個人で利用できる常時接続環境は今後どうなっていくのでしょう？ フレッツ・ISDN、xDSL、CATVインターネット、高速無線接続、光ファイバなど、さまざまなサービスが群雄割拠するといった状態になりそうです。

安い・ハヤい・ウマいの3拍子そろったサービスが提供されるのであればどれでもいいんですけどね。

今月は連載特集の第2回をお届けしました。いかがでしたでしょうか？

あとで読んでも楽しめます?!

Windows党だったはずが、夏ごろからLinuxにはまり始め、今ではもう抜け出せないぐらいどっぷり浸かってしまいました。貴誌の記事は最近ようやく

理解できるようになりました。バックナンバーを改めて読み直すと、昔わからなかったことがだんだん理解できている自分にうれしくなっています。

（Stardustさん）

LinuxはWindowsに比べると設定する項目も多く、設定の仕方も統一されていません。そのため、最初はとまどうことも多いのですが、慣れてくると一気に面白くなります。

Windowsを使っていると、できることが増えていく楽しさはなかなか味わえませんが、Linuxでは確かな手応えを日々感じることができます。

バックナンバーをお持ちの方は、ときどき読み返していただくと新たな発見があるかもしれません。

ハガキからWebになりました

1月号からプレゼントコーナーはハガキではなくWebになったんですね。このほうがわざわざハガキを出しに行かずにいいのですが、『Linux magazine』を買ってない人もプレゼントに応募できてしまうような気がするのですが……。

（國枝信吾さん）

1月号からアンケートハガキがなく、忘れたのかと思い、苦情を書こうとしましたが、このような形態になったのですね。びっくりしました。Linux magazineは読みやすいので、今後も購読していこうと思っています。

（Kosodoroさん）

いつも送らせていただいてます。今回はちょっと厳しい意見を言わせてください。1月号からWebで応募できるようになったのですね。たしかにこのほうが便利ですが、本誌を買わないで懸賞目的で応募する人が増えると、毎号買って応募してた私たちにとっては、当選の確率下がり、不利になるだけな気がします。以前のマークシートも面倒だけど好きだったんですが……どうか元に戻してください。

（細川誠さん）

急な変更でいろいろとご迷惑をお掛けしました。読者様以外もプレゼントコーナーへ応募できてしまう件は、今月号より簡単なクイズにお答えいただくことで対応させていただきました。

みなさまよりのメッセージをリアルタイムに読むことができるようになりましたので、『読者の声』担当は、毎日何度もアンケート結果をチェックするのが習慣になりました。たくさんのご意見、ご感想をお待ちしておりますので、ぜひともアンケートへご協力ください。

Linuxは自由なOSです

本当ならサーバ構築とかしないといけないのでしょうが……。デスクトップテーマを色々とダウンロードし、画面を見てはニンマリしている今日この頃。初心者なんだからこんなのもアリですよネ！　そろそろマジメにLinuxの勉強を再開しないと〜。どうせ長続きしないとタカをくくってジャンクマシン（Pentium 100MHz）でチマチマと遊んでいましたが、NEWマシンが欲しくなってしまった。ウルウル。

（田辺 斉さん）

Linuxに限らず、面白そう！　という分野からいろいろとやってみるのが上達への近道です。楽しく遊んでいると、いつの間にかいろいろな知識が身についていて、新しいマシンやパーツが部屋に増え……。ハマりすぎにだけはお気をつけください。睡眠不足と金欠病に悩まされるようになったら、すでに危険な状態です。

それにしても、海外で作られるデスクトップテーマはおどろおどろしいものばかりなのはなぜなんでしょうね？

Emacsはひとつの宇宙です

1月号の『Emacsはじめました』は、とてもありがたかった。TeX／LaTeXにはじめて触れたのが1992年だから、もう8年になる。最初はFM-TOWNSのDOS-Extender上で動作するもので、Windows3.1／95でも利用し、今はVine Linuxを中心に愛用している。Emacsを使い始めて2年余りになるが、TeX／LaTeXで文書を作成編集する者には、手放せない環境だ。今回の記事の内容で、知らなかったことがいくつかあった。マウスを使わないで実用的なオペレーションができ、嬉しい。どうもありがとうございます。

（石垣立郎さん）

担当はＥＰＳＯＮの２８６マシンにCx486SLCと8Mバイトのメモリを載せてDOS-Extender版のLaTeXを使っていました。今では考えられないほど非力な環境でしたが、文書の作成には大活躍してくれました。

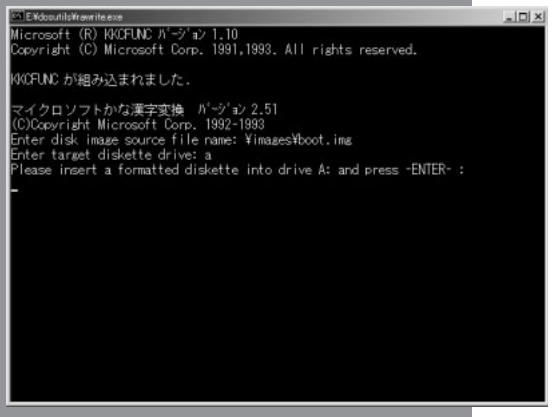
Emacsの達人は、一度Emacsを起動したらすべての作業はそこで行い、マシンの電源を落とすときまで終了しないものだと聞きます。それくらい快適な環境だということなんですね。編集部のメンバーはEmacs派とvi派に分けられますが、担当は両方とも使いこなせないので、どちらの派閥にも入れてもらえません。

付録CD-ROMに収録した

LASER5 Linux 6.4のインストール

本誌付録CD-ROM収録のLASER5 Linux 6.4はFTP版です。非商用ソフトだけが含まれており、製品版を販売しているレーザーファイブのサポートを受けることはできません。

また、CD-ROMのメディアに不良があった場合は、お手数ですが（linux-cd@ml.ascii.co.jp）宛にご連絡くださるようお願いいたします。



ブート用フロッピーディスクの作成

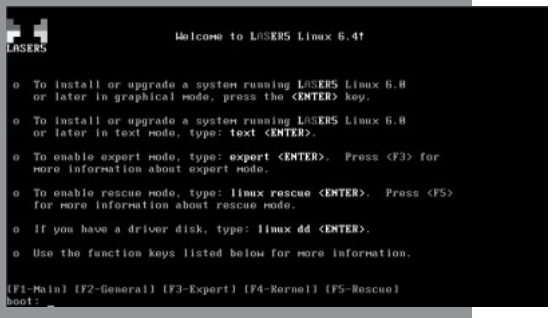
インストールするマシンがCD-ROMから起動できる場合は、CD-ROMブートでインストーラを起動します。CD-ROMから起動できない場合は、以下の手順で、インストーラ起動用のフロッピーディスクを作成します（ここでは、フロッピーディスクドライブがA:であるとして解説します）。

（1）Windowsのエクスプローラで、CD-ROMの「dosutils」というフォルダを開き、その中にある「rawrite」をダブルクリックします。

（2）DOS窓が開き、ファイル名の入力を促してくるので、

\images\boot.img

と正確にタイプして[Enter]を押します。さらに、フロッピードライブ名を求めてくるので、Aをタイプして[Enter]を押します。最後にフロッピーがセットされていることを確認して[Enter]を押すと、フロッピーの作成が始まります。



インストーラの起動

作成したフロッピーディスクや、CD-ROMをドライブにセットして、マシンを再起動します。インストーラが起動し「boot:」というプロンプトが表示されたら[Enter]を押します。

しばらくすると、Xを使ったグラフィカルな画面が表示されます。グラフィカルなインストール画面がうまく表示されない場合は、「boot:」の箇所で「text」とタイプして[Enter]を押せば、テキスト画面のインストーラが起動します。



使用言語とキーボードの設定

日本語を使う場合は、デフォルトのまま「Japanese」を選択して「次」を押します。

次はキーボードの設定です。デフォルトで「モデル」に「Japanese 106-key」、「レイアウト」に「Japanese」が選択されています。日本語キーボードを使用するときはデフォルトの状態で「次」を押します。

4

マウスの設定

PS/2タイプの2ボタンマウスを使う場合は「2 Button Mouse(PS/2)」をチェックします。「3ボタンマウスのエミュレーションを設定する」は、2ボタンマウスの左右ボタンを同時に押すことで、3ボタンマウスの中ボタンを代用する機能です。これは便利な機能なので、チェックしたまま「次」を押します。



5

インストールタイプの選択

「GNOME Workstation」と「KDE Workstation」を選択すると、ブートローダLILOがMBRにインストールされます。LILO以外のブートローダを使用する場合や、Windows NT／2000を使用する場合は、これらのタイプを選択してはいけません。ここでは、柔軟なオプション選択ができる「Custom」を選択して「次」を押します。「アップグレード」は、すでにLASER5 Linuxの以前のバージョンをインストールしているユーザー用です。



6

パーティションの作成

Linuxをインストールするには、最低限Linuxシステム用とSwap用の2つのパーティションが必要です。「追加」を押すと、画面のようなダイアログが表示されます。Linuxシステム用のパーティションは、「マウントポイント」に「/」を、「パーティションタイプ」は「Linux native」を選択します。Swap用パーティションは「Linux swap」を選択します。各パーティションサイズは、以下を目安にして設定してください。

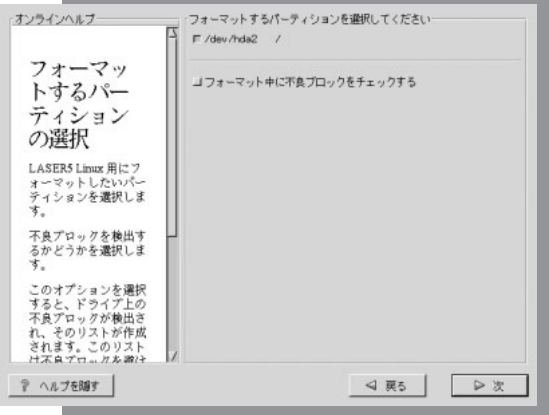
Linuxシステム用　：フルインストールの場合は2Gバイト程度以上
Swap用　：搭載メモリの1～2倍程度

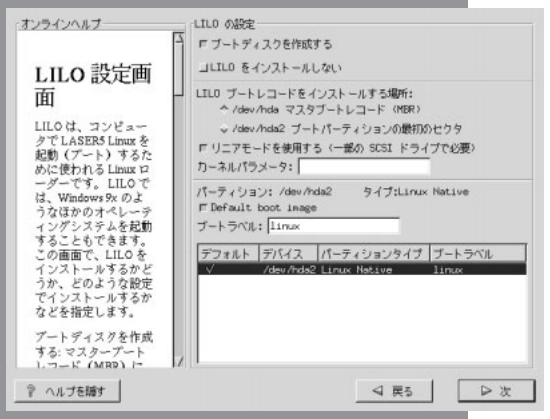


7

パーティションのフォーマット

前の場面で作成したパーテシションをフォーマットします。チェックされている「/dev/hdax」パーティションがフォーマットされます。「フォーマット中に不良ブロックをチェックする」を選択しておくと、パーティションのフォーマット中に、ディスクの不良箇所を調べてくれます。最後にもう一度フォーマットするパーティションを確認して「次」を押します。





LILOの設定

「LILOブートレコードをインストールする場所」は環境に応じて選択します。

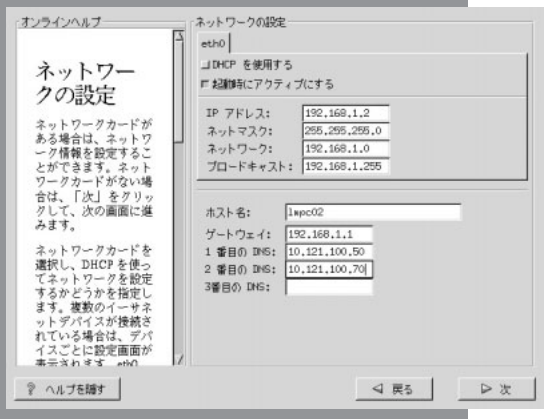
/dev/hdaマスターブートレコード（MBR）

- LILOを使用してWindows 9xとLinuxを起動時に選択する
- ディスクにはLinuxのみをインストールする

/dev/hda2ブートパーティションの最初の（第1）セクタ

- System Commanderなど、LILO以外のブートマネージャを使用する
- Windows NT／2000とLinuxをデュアルブートする（ブートディスクが必要）

緊急時のために「ブートディスクを作成する」をチェックしておくとよいでしょう。



ネットワークの設定

家庭内で、ISDNルータやほかのサーバマシンでDHCPサーバを稼動させている場合は、「DHCPを使用する」を選択します。DHCPサーバがない環境でLinuxを使う場合は、以下のようなネットワークアドレスを入力するとよいでしょう。

IPアドレス	192.168.1.2～192.168.1.254
ネットマスク	255.255.255.0
ゲートウェイ	家庭内サーバマシンのIPアドレス
1番目のDNS	プロバイダなどのDNSサーバのIPアドレス



タイムゾーンの設定

タイムゾーンの設定は、デフォルトで「Asia/Tokyo」が選択されています。マシンを日本で使うときは、このままの状態で「次」を押します。



ユーザーアカウントと認証の設定

Linuxを使用するユーザーを設定します。まず、Linuxシステムに必須の管理者のパスワードとして、「Rootパスワード」と「確認」に同じものを入力します。管理者のユーザー名は「root」です。

次に一般ユーザーのアカウントを設定します。「アカウント名」、パスワードとして「パスワード」と「パスワード（確認）」に同じものを、「フルネーム」にユーザーのフルネームを入力して、「追加」を押します。この要領で複数の一般ユーザーを作成できます。ユーザー設定を終えたら「次」を押します。

認証の設定場面はデフォルトの状態で「次」を押します。

パッケージグループの選択

おおざっぱなパッケージグループを選択します。パッケージ群の意味がわかるユーザーは、好みのパッケージ群を選択し、どのパッケージ群を選んでよいかわからないユーザーはすべてのパッケージをインストールするために、最下段にある「Everything」を選択して「次」を押します。

12



Xの設定

モニタの一覧が表示されます。リストに使用するモニタがない場合は、使用するモニタに無理のないクロックのものを「Generic」の「Generic Multisync」から選択します（クロックの選択についてはモニタのマニュアルを参照してください）。

次にXサーバの設定です。ほとんどのビデオカードは自動で認識されます。色数や解像度を手動で設定する場合は「X設定のカスタマイズ」をチェックし、グラフィカルなログイン画面を使う場合は「グラフィカルログイン」をチェックして「次」を押します。

13



パッケージインストールの開始

画面（手前）で「次」を押すと、パッケージのインストールが始まります。インストールが終わるまでしばらく待ちます。

14

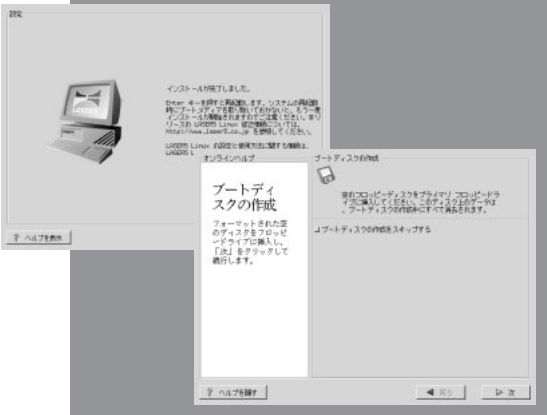


インストールの終了

LILOの設定場面で「ブートディスクを作成する」をチェックした場合は、手前の画面が表示されます。「次」を押すと緊急時用のブートディスクの作成が始まります。フロッピーの作成が終わると、奥の画面が表示されます。フロッピーディスクを抜いて「終了」を押します。

以上でインストール作業は終わりです。お疲れさまでした。

15



付録CD-ROMに収録した

Plamo Linux 2.1のインストール

(1) Plamo Linuxのインストーラは、ビデオカードを自動認識しません。また、ネットワークカードも自動認識しない場合があるので、インストール前に、あらかじめマシンのハードウェア情報を調べておく必要があります。
(2) 以下のインストール解説はPC/AT互換機用です。NEC PC-98x1マシンへのインストールについては、Plamo Linuxのページ（http://www.linet.gr.jp/~kojima/Plamo/）などから得られる情報をご参照ください。

1 インストーラ起動用フロッピーの作成

CD-ROMブート可能なマシンでは、CD-ROMからインストーラを起動します。CD-ROMブートができないマシンの場合は、インストーラ起動用のフロッピーディスクを作成します。手順は以下のとおりです。

まずCD-ROMと、空のフロッピーディスクをマシンにセットします。次に、コマンドプロンプトを開き、以下のように操作します（ここではCD-ROMドライブ名をD:やQ:としましたが、マシン環境に合わせて入力してください）。

■PC/AT互換機の場合
（1）d:[Enter]
（2）cd at\install[Enter]
（3）rawrite \boot\bootdsk a[Enter]

■NEC PC98x1の場合
（1）q:[Enter]
（2）cd 98\144[Enter]
（3）makeplfd[Enter]
（4）デスクトップ用だと「3」を選択

2 インストーラの起動

作成したフロッピーディスクかCD-ROMをセットして、マシンを起動します。インストーラが起動して、しばらくすると「plamo login:」と表示されるので、「root」と入力して[Enter]を押します。さらに「setup」と入力したあとで[Enter]を押せば、Plamo Linuxのインストールが始まります。



3 キーボードの設定

画面に設定メニューが表示されます。まず「KEYMAP」を選択して、キーボードの設定を行います。メニュー画面の操作方法は、矢印キーや[Tab]で選択し、[Enter]で次へ進みます。「KEYMAP」を選択すると奥の画面が表示されます。日本語106キーボードを使用する場合は、デフォルトのまま<OK>でよいですが、CTRLキーとCapsLockを入れ替えたり、[全角／半角]キーを[ESC]に割り当てるときは、必要な項目に[Space]キーで[X]とチェックを入れて設定します。



パーティションの作成

Plamo Linuxではパーティション作成時にfdiskというツールを使います。ここでは、ハードディスクがプライマリマスタに接続されているものとします。画面（手前）のように「fdisk」を選択して<OK>とすると、Linuxをインストールするディスクの選択画面になります。Linuxでは、プライマリマスタが/dev/hdaとなるので、/dev/hdaを選択して<OK>を押します。



fdiskの操作

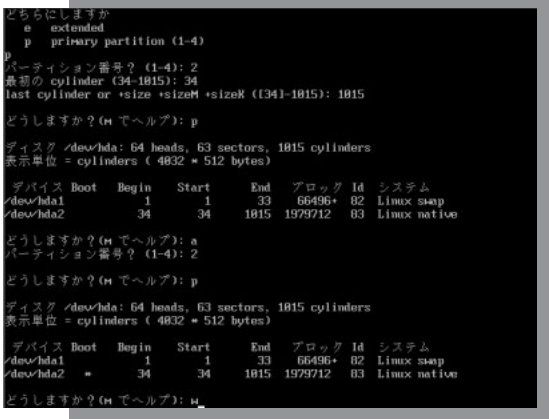
Linuxをインストールするには、システム用とSwap用の2つのパーティションが必要です。fdiskというツールを使い、パーティションを作成します。

■Linuxシステム用パーティションの作成

nコマンドを用いて、Linuxシステム用パーティションを作成します。お勧めパッケージをインストールする場合は、800Mバイト程度確保すればよいでしょう。

■Swap用パーティションの作成

nコマンドで物理メモリの1～2倍程度のパーティションを作成します。次にtコマンドを使い、パーティションIDを82にセットします。pコマンドを使い、システム用とSwap用の2つのパーティションが作成されたことを確認します。最後にwコマンドでパーティション情報を書き込みます。



インストール開始

それではPlamo Linuxのインストールを始めます。メニューから「install」を選択して<OK>を押します。



Swapパーティションの設定

すでにキーボードの設定は終わっているので、インストール後にSwapパーティションが使えるように、上から3番目の「ADDSWAP」を選択して<OK>を押します。インストーラがSwap領域を表示するので、<Yes>を押します。このあとで、いくつかの質問が続きますが、「Linuxをインストールするパーティションの選択」まで[Enter]を叩いていけばよいでしょう。





システムパーティションの作成（その1）

今度はLinuxシステムをインストールするパーティションの設定です。Linuxシステム用として使用できるパーティションが表示されるので、リストの中から1つ選択して<OK>を押します。有効なパーティションが1つしか表示されない場合は、そのまま<OK>を押します。

次にLinuxシステム用のパーティションをフォーマットします。選択するのは「Format」か「Check」のどちらかです。ハードディスクのフォーマットのみ行う場合は「Format」を、ハードディスクに異常がないか確認しながらフォーマットする場合は「Check」を選択して<OK>を押します。



システムパーティションの作成（その2）

ファイルシステムの単位であるi-nodeは、デフォルトのまま「4096」を選択して<OK>を押せばよいでしょう。



インストール元の選択

インストールするメディアを選択します。ここでは本誌付録のCD-ROMを使っているので、「CD-ROMからインストール」を選択して<OK>を押します。次にマシンに接続されたCD-ROMドライブを探すため、「scan」を選択して<OK>を押します。



インストール方法の選択

インストーラがCD-ROMドライブを探し出すと、画面のように表示されます。次の「インストール方法の選択」では「plamo」を選んで<OK>を押します。

パッケージの選択

インストールパッケージの選択です。デフォルトで「お勧めパッケージセット」がチェックされているので、そのまま<OK>を押します。次の画面では、インストールするマシンがノートPCかデスクトップか、使用する日本語入力プログラムはCannaかWnnかで、メニューが分かれます。デスクトップマシン上でCannaを使う場合は、右の画面のように「deskc」を選択して<OK>を押します。

12



パッケージインストールの開始

<Yes>を押して、選択したパッケージのインストールを始めます。インストール状況が表示されたら、終了するまでしばらく待ちます。

13



Xサーバのインストール

Plamo Linuxはビデオカードを自動認識しません。マシンを開けてビデオカードのチップを調べるか、Windowsを使ってビデオカード情報を調べておきます。デフォルトではどんなビデオカードでも使用できる「16色用 Xサーバ」がチェックされています。これに加えて、Xがきれいに表示できるように、ビデオカードに対応したXサーバにチェックを入れて<OK>を押します。このあと、しばらくの間パッケージインストールが続きます。

14



カーネルのインストール

「お勧めパッケージセット」を選択していれば、Linuxカーネルは自動でインストールされているので、「skip」を選択して<OK>を押します。

15





起動用フロッピーディスクの作成

パッケージのインストールがひと通り終ったので、各種設定に入ります。「システム設定」で<Yes>を押して次へ進みます。緊急時に使う、Plamo Linuxの起動ディスクを作成します。まず、空のフロッピーをセットし、「format」を選択して<OK>を押します。フロッピーのフォーマットが終了したら、「simple」か「lilo」を選択して<OK>を押します。ここでは「simple」を選択して、Linuxカーネルのみをフロッピーへ書き込みます。起動用フロッピー作成が終わったら、「continue」を選択して<OK>を押します。

16



モデムの設定

ダイヤルアップ用にモデムを使う場合は、「モデムの設定」で<Yes>を押します。次の画面では、モデムをどのポートに接続しているかを選択して<OK>を押します。多くのマシンでは「ttyS0」か「ttyS1」のどちらかでしょう。

17



マウスの設定

PS/2タイプのマウスを使う場合は、画面のように、上から2番目を選択して<OK>を押します。

18



LILOのインストール（その1）

複数のOSをマシン起動時に選択するブートマネージャLILOをインストールします。ここではデフォルトのまま「simple」を選択して<OK>を押します。

19

LILOのインストール（その2）

LILOのインストール場所を選択します。System CommanderなどLILO以外のブートマネージャを使うときや、LinuxをWindows NT／2000とデュアルブートするときは「Root」を選択し、Windows 9xとデュアルブートするか、マシンをLinux専用に使う場合は「MBR」を選択します。「Floppy」を選ぶとLinuxの起動は遅くなりますが、OSの起動で問題が出ないので安全です。

20



ネットワークの設定

この場面では、DHCPを利用するための設定が行えないので、IPアドレスなどの値を直接入力します。モデムやTAを用いてネットワークを使用する場合は、<No>を押してスキップします。ネットワークカードを使用する場合は、<Yes>を押して以下の項目を入力していきます。

- ホスト名
- ドメイン名
- IPアドレス
- ゲートウェイ
- ネットマスク
- DNS

21



タイムゾーンの設定

日本でPlamo Linuxを使うときは、デフォルトのまま「Japan」を選択して<OK>を押します。

22



終了

インストールが終了しました。[Ctrl]、[Alt]、[Delete]を同時に押して、マシンを再起動します。お疲れさまでした。

23

