教育用大規模計算機システムにおける管理の省力化手法

齊藤明紀†

教育用の計算機システムは，一般に利用者数，および，管理すべき計算機の台数の両方が多いために，システム管理の負荷が大きい．さらに，利用者が移動すること，広範囲に分散配置されていること，24 時間稼働していること等も問題となる．本稿では，このような多人数が利用する，多数の計算機をネットワークで接続したシステムの維持・管理を省力化するための方法として，多数の計算機の初期設定や設定変更等を集中管理で行うパッチシステムおよびそれを有効に稼働させるためのいくつかの手法をを提案する．パッチシステムでは，設定変更や更新すべきアプリケーションプログラムをパッチ・スクリプトの形でサーバが提供し，管理対象端末で動作する管理プログラムが必要に応じてパッチを端末自身に適用する．パッチシステムの耐故障性を向上させるため，最低限のシステム設定情報は個々のローカルディスク上に置いている．これら設定ファイルの内容は，パッチシステム自身によって更新される．

Maintaning Campus-wide Educational Computer System

Akinori Saitoh†

Campus-wide educational computer system is consist of many personal computers or workstations. They have enormous users registerd and location of computers are spread over university campus. However maintaining such systems imposes heavy workload to keep systems well-managed. We have shown several methods for reducing such workload on such system administration for campus wide educational systems. To reduce those workload, we propose a method called “patch-system” and related subsystems for configuration management and application program remote installation. On the system, system configuration modification information and application binary updata are registerd on a center server as a series of patch script files. Each terminal autonomously refers to server to find new patch to apply. To improbe fault tolerancy of the patch-system for network trouble, minimum administrative information are stored on each local disk. Those files are maitained through patch-system itself.

1. はじめに

本稿は，大阪大学情報処理教育センター☆で 1996 年 3 月から 2000 年 2 月まで利用した NEXTSTEP を端末 OS とする情報教育用電算機システム[1),2)] の管理運用を行った際に用いた方式[3)] のおよび運用上得られた知見についてまとめたものである．

大学等における教育用計算機システムとして，多数のパソコンやワークステーションをネットワークによって接続した分散型のシステムが多く利用されるようになってきた．

現在ではこのような教育用ネットワークは IP ベースの技術を用い，インターネット/イントラネットとして構築することが普通になっている．そこでは，IP をはじめとして NFS，NIS，SMTP，POP/IMAP，NNTP 等オープンな標準規格や業界標準規格を用いて，既存のツールやアプリケーションを組み合わせるだけで相当の応用システムが構築できるようになっている．

しかし，システム全体が正常にサービスを提供し続けるための維持管理作業はシステマチックなサポートは用意されておらず，このような教育用計算機の環境をつねに良い状態に保ってシステムを維持管理するには，非常に多くの手間がかかる．

2. 問題点と目標

大学等の学生全体を対象とした教育用計算機システムでは，利用者数が多いこと，および，管理すべき計算機の台数が多いために，上記の作業負荷が大きくな

† 大阪大学大学院基礎工学研究科情報数理系専攻
Graduate School of Engineering, Osaka University

☆ 現在ではサイバーメディアセンターに改組されている．

る．さらに，これらの作業に加えて，環境の特殊性を考慮しなければならない．

まず，端末の配置が地理的に分散していることがある．たとえば本センターでは 2 つのキャンパス全体にわたって各学部に分散端末室を設置しており，端末室まで保守作業のために訪れること自体が運用上の重大な問題点となっている．

このようなシステムでは会社等の個人用パーソナル・コンピュータとは異なり，利用者自身にシステム管理の一端（OS パッチの適用等）を担わせることができない．一部の運用スタッフで全体を管理運用する必要がある．

本稿では，多人数が利用する，多数の計算機をネットワークで接続したシステムの維持・管理を省力化するためのいくつかの方法を示し，その有効性を明らかにする．本稿で取り上げるのは，管理業務のうち特に端末管理の省力化の手法である．

本稿で取り上げる管理方式の設計の基本方針は以下のようなものである．

- 端末の管理作業の発生を抑止することを目標とする．
- 人手による管理作業をできるだけ避け，自動化する．
- できるだけネットワークごしに遠隔管理を行う．全端末を巡回するような作業が発生しないようにする．
- 削減しきれない人手による管理作業は，難易度を軽減し，できるだけ簡便な作業にする．

さらに，以下の前提条件に留意して設計されている．

- 本質的な省力化を実現する．
  たとえば，更新導入する新システムにシステムインテグレーション（SI）や運用サービスを含める等することにより省力化をはかることはできる．しかしこれは作業担当を大学スタッフから業者に移しているだけで実質的には省力化ではない．
- ソフトウェア的トラブル等で稼働していない（ハングアップしている）端末や故障や停電で電源が切れている端末が常時存在することを想定する．「全端末が電源を投入され正常に稼働している瞬間」を要求するような管理方式は採用しない．

3. パッチシステム

多くの端末を一元管理する場合，構成管理は重要な問題である．アプリケーションの追加や各種設定変更等は，台数に比例した作業量になりがちであり，多数の計算機を分散配置したシステムの維持管理の作業人員のピーク需要の要因である．

構成管理は，ファイル配布ツールを用いて設定ファイルやアプリケーションファイルを管理対象に配布することによって行うことが多い．我々は，ファイルの内容ではなく，各種設定を行う手続き（スクリプト）を情報サーバで提供し，各端末でそれを実行することで設定内容の更新を行う方式を採用した．これを “パッチシステム” と称している．

ここで紹介するパッチシステムは，以下の目的のために使用している．

- 多数の計算機の個別の設定を集中制御する．
- 故障等で HDD を交換したときの再設定を行う．
- 各端末の状況（利用中，停電中等）に関係なく作業できる，すなわち，深夜や休日等，利用者が少なくなる勤務時間外の作業を作業を強いられない．

このパッチシステムの原型は筆者が大阪大学基礎工学部情報工学科の計算機演習室システムの構成管理のために開発したものである．管理対象台数の増加と 24 時間運転の分散端末室の存在に対応して機能を拡充した．

本パッチシステムの特長は，端末側が自律的に動作して情報サーバを参照するプル型サービスであるところにある．多くのシステム管理ツールは，中央の管理サーバが端末にポーリングを行うプッシュ型サービスとして実現されている．プッシュ型の場合，管理対象が大規模になると故障等種々の理由で通信できない端末の存在が無視できなくなる．また，ハードディスクの故障の際等設定状況が過去に戻る場合は，それを検出して送付済みのファイルを再度送付しなければならない．

さらに，端末がネットワークごしの制御を受け付けない状態になった場合は，プッシュ型のシステムは対処できない．対象台数が非常に多くなると，動作負荷も問題になってくる．

3.1 パッチによる構成管理

各端末は，「パッチレベル」の値をローカルファイル（`/LocalAdmin/conf/PATCHLEVEL`）に持つ．設置当初はパッチレベル 0 である．パッチファイルおよび，「現在のパッチレベル」情報はサーバから NFS で提供される．ファイル名は setup.{ パッチレベル } となっており，`setup.`n には，端末のパッチレベルを $n-1$ から n にする手続きが格納されている．

現在の自身のレベルとサーバに登録されたレベルを比較し，違っていれば対応する setup.n を順次実行する．このとき，何らかのエラーが起きればその時点で以降のパッチファイルの実行をとりやめる．エラーを

```
#! /bin/sh
#SINGLEUSER
# clear above line if this can do in MUITIUSER mode
VERSION=25
# by toyonaka!saitoh   ttyp3   Apr 14 04:38     (tgray15)
# by Sat Apr 20 20:35:02 JST 1996
while read DIR file ;do
export DIR
(cd $DIR && tar xvfp -) < files.$VERSION/$file || exit 2
done <<EOF
/ develop.tar
/LocalAdmin/conf fstab.tar
/LocalAdmin hourly.tar
/tmp rc.tar
EOF
[ $? -eq 0 ] || exit 2
[ -f /tmp/rc ] || exit 2
mv /etc/rc /etc/rc.sav
mv /tmp/rc /etc/rc

case `hostname` in
toyonaka*|ecipt[0-9]*)                          code=tc;;   #toyonaka center
suita*|ecips*)                                  code=sc;;   #suita center
jinka*|igaku*|hoken*|sigaku*|yakugaku*|kougaku*) code=sb;;   #suita bunsan
*)                                              code=tb;;   #toyonaka bunsan
esac
cp /LocalAdmin/conf/fstab.$code /etc/fstab.new || exit 2
rm -f  /etc/fstab.sav
ln /etc/fstab /etc/fstab.sav || exit 2
mv /etc/fstab.new /etc/fstab || exit 2
ln -s /Net/toyonaka/home-toyonaka /home-toyonaka
ln -s /Net/suita/home-suita /home-suita
niload -d -r -t /mounts localhost/local <  /dev/null
niload  -t fstab localhost/local < /LocalAdmin/conf/fstab.neti.$code
chmod 751 /private/spool
#
sync; sync; sync
echo $VERSION > /LocalAdmin/conf/PATCHLEVEL
sync; sync; sync
fbshow -B -I "Configration changed to $VERSION. Rebooting..." -z 47
reboot
```

図 1 パッチファイルの例
Fig. 1 Patch file example.

起こしたレベルのパッチは，次回のブート時あるいは定時チェック時刻（5 章）に再度行われる．エラーが起こって中断された setup.n は次回再度実行されるので，それを考慮した書き方をしなければならない．

パッチスクリプトの実行エラーが繰り返し起こっている端末は，パッチレベルを調べて検出し，個別に対応する．このような端末は存在しても例外的な少数なので，運用の手間の点では大きな問題とはならなかった．

図 1 はパッチファイルの例（setup.25）である．第 2 行の `SINGLEUSER` は，この作業がリブートの過程（利用者がログインしていないとき）にしか行えないことを表す．`setup.25` の場合は，fstab の更新にともない最後に再起動が必要となるためである．

シェルスクリプトを用いるため，設定の記述能力は十分にある．たとえば，`case` 文や `if` 文を使うことで，個別に異なる設定が可能になる．この例でも，設置した地区（豊中，吹田）とセンター教室か分散端末室かを case 文で判断して，それぞれ異なる fstab を適用している．

3.2 個別化設定の分離

システム導入後 1〜2 年経過すると，プリンタの移設や端末の移設で個別設定をやり直す必要が生じるようになる．当初のパッチシステムでは，全端末に対する共通の設定変更やソフトウェア配布と個別の設定項目（たとえばパラレルポートにプリンタが接続された端末だけの設定や，吹田地区特有の設定）が個々のパッチスクリプトの中に混然一体となって記述されていた．

このため，プリンタの移設（接続した端末を変更）といった作業だけでも，パッチレベルを人為的に 1 まで戻すか，状況をよく知ったエキスパートが必要な変更を手動で行うといった，時間や手間がかかる手法をとる必要があった．

そこで，パッチスクリプトから，端末個別の設定作業を別ファイルに括り出した（**図 2**）．

たとえばファイルサーバの構成変更が行われて NFS クライアント設定（fstab）を変更する必要が生じた場合を考える．当初のパッチスクリプトでは図 1 のよう

```
#!/bin/sh
sh setup.hosts || exit 1
sh setup.crontab || exit 1
sh setup.homelink || exit 1
sh setup.mounts || exit 1
sh setup.snmp || exit 1
sh setup.ntp || exit 1
sh setup.printcap || exit 1
```

図 2 個別化再設定スクリプト
Fig. 2 Reconfiguration script.

に fstab を書き換えるコマンドを直接記述していた．これを，fstab を端末ごとの設定に書き換えるスクリプト setup.fstab として括り出し，パッチスクリプトでは setup.fstab を呼び出すようにした．

これにより，端末の個別化設定だけやり直すにはホスト名変更後に括り出された個別化設定スクリプトを順に実行すればよくなった．

4. 初期設定の構築

計算機の設置調整は最も集中的に労力を必要とする工程である．

我々は，パッチシステムを前提にすることで，設置調整を効率的に行うことができた．また，端末機の納入業者が大学のネットワークに接続することなく動作確認を行えるように配慮した．

教育用の演習室では，すべての端末計算機で同様の環境を提供することを目的とする．そこで，各計算機（NEXTSTEP 3.3J を搭載した AT 互換機）の設定はほぼ同じであるが，ハードウェア構成は，

- MO/CDROM の有無，
- ローカルプリンタの有無，
- セカンド Ethernet カードの有無（ゲートウェイモデル）

という差異がある．ソフトウェア的には，ホスト名，IP アドレス，使用できる（同一教室内に存在する）プリンタ，経路情報，使用するファイルサーバ等を個別に変更しなければならない．

OS のインストールおよび個別設定作業を多く（本システムの場合は 500 台）について繰り返すのは非常に手間がかかる．そこで，以下のような設定の端末計算機の雛型（マスタディスク）を大学側で作成し，工場ではその複製を生産するように依頼した．

- ハードウェア構成の違いにかかわらず，ハードディスクの内容はまったく同じである．
  当初設定（生産した状態）ではネットワークやサーバ計算機に依存せずにブートし，単独で動作確認が可能となっている．
- 最低限の個別設定機能を持つ．

このマスタディスクは，必要なオプションソフトのインストールや不要なソフトウェアの削除，およびセキュリティ対策等事前に必要性が予想できたカスタマイズはすべて済ませたものである．

マスタディスク作成時には，実運用状態で必要な設定のすべてが予想できるわけではない．そこで，初期設定プログラムでは，ネットワーク関係の設定を最低限に初期化して，パッチシステムが動作するようにすることを主目的としている．

各教室に端末計算機を設置したときの個別作業は，以下のようなものである．

- ネットワークケーブルを接続せずに単体で起動することを確認し，
- ネットワークに接続して再起動し，
- 日時（カレンダ）情報とホスト名と SCSI の有無を入力する．プリンタの有無は入力しない．これは，実際に設置した際の実態に合わせてサーバに登録し，後日パッチシステム経由で設定した．

ホスト名を受け取るのはあらかじめ搭載した初期設定プログラムで，自機 IP アドレスを算出し，サーバへの経路情報を設定する．端末は初期設定後は稼働状態のまま（電源を切らずに）置かれ，パッチシステムによって個別設定が継続される．この段階では，ログイン可能なのはシステム管理用ユーザのみであり，利用者用ファイルサーバもマウントはしていない．

ハードディスクの初期設定をすべて同じにできたため，工場での組立て時にハードディスクの物理コピーツールで OS 一式をインストールすることが可能となった．

また，輸送の際にもどの部屋に設置する端末かを意識せず，単に設置台数だけを数えて輸送することができた．このようにして，工場での生産だけでなく出荷・輸送・搬入作業も効率化された．設置後の作業も，個別に OS を導入すれば数十分を要するものが数分に短縮された．

4.1 パッチによる設定回復

ハードディスクの故障や復旧不可能なファイルシステムの損傷で，端末の OS の再インストールがある程度起こるのは避けられない．通常の保守契約ではハードウェアの交換とせいぜいファームウェアレベル（BIOS 設定）の再設定までしか行われない．ソフトウェアの細かい再セットアップは大学側の作業となる．これは煩雑な作業であるだけでなく作業の発生が予測できないため，運用上の負担は大きい．

パッチシステムを前提とすると，OS の初期インス

トール状態が 1 種類でよいため，修理用の予備ハードディスクにあらかじめマスタディスクからコピーしておけば，交換するだけで修理を済ませることができる．その後は当初の設置時と同じく時刻とホスト名を入力してしばらく待てば，その時点での正しい状態にまで自動的に設定される．

マスタディスク製作には手間がかかる．本システムの場合約 1 人月程度の手間を要したが，4 年間のシステム運用期間を通じて単一のマスタディスクを使用し続けることができたので全体通じてみると労力の削減が実現できたと考えられる．

5. 定時自己診断

パッチシステムによって各種の設定変更が自動化されるが，場合によってはシステムの再起動が必要となる場合もある．たとえば，NFS のマウントパラメータの変更がそうである．しかし，新しいパッチが発行されたからといって利用者がログインして作業している最中に再起動してしまうことは望ましくない．

再起動しても構わない状況かどうかを中央サーバからのポーリングで調べることは効率が悪い．またネットワークごしの遠隔操作での再起動はつねに可能だとは限らない．たとえば，ファイルサーバで何らかの作業を行った結果 “stale NFS file handle” 等のエラーが端末側で起こってログイン不能になることがある．

一方，端末の障害回復の手段として再起動を行うこともある．サーバ機では停止の与える影響が大きいので，再起動はできるだけ避けるが，端末では障害の追求ために人手をかけることができないため，再起動を第 1 に試みることが多い．

そこで，各端末で定期的（1 時間に 1 回）にパッチシステムが動作する際に，簡単な自己診断も行って，必要に応じて再始動するようにしている．

基本的にはログイン中のユーザの使用継続を優先することで，実質的可用性の向上を実現している．

調査項目は以下のとおりである．

- ファイルサーバにアクセスできるか？ ディレクトリは参照できるか？ ファイルの書き込みアクセスは成功しているか？
- スタンドアローンモードで動作中か？
- パッチシステムで，再起動が必要な未適用パッチが存在するか？
- パッチスクリプトの異常終了ははなかったか？
- 前回の自己診断プロセスがハングアップしているか？

何らかの障害が発見された場合はさらに以下の点を調べる．

- ファイルサーバおよびネットワークが正常か？ rup コマンドでサーバが確認できない状況では，再起動よりもサーバ（への通信）の回復を待つ方が適切である．
- 障害の深刻度は？
- 障害が何回（何時間）連続して起こったか？
- 現在ログインしている利用者はいるか？

これらを総合して，現在の利用者がログアウトした際にリブートするよう予約するか，再起動を行う（shutdown -r）かいずれかの対処を行う．

自己診断作業そのもの異常に対処するため，まず最初にエラー回数のカウンタを進め，エラーがなければ最後にカウンタをリセットしている．

たとえば，パッチのための再起動が必要という軽度の場合には，ログインしている利用者がいる状態では再起動しない．軽度の再起動必要性が 3 回起こると，中度と見なし「ログアウト時再起動」を予約する．5 回（5 時間連続）起こると，重度と見なし，利用者がログインし続けている場合でも，再起動する．これは，ログアウトし忘れて放置された端末への対策でもある．一方 NIS が動作していないという重度の障害がありかつサーバが動作しているときは，1 回めの検出で再起動する．

以上の作業は 1 時間に 1 回ずつ各端末で行われる．多くの端末が同時にサーバにアクセスすることは望ましくないため，ホスト名や IP アドレスを用いた簡単なハッシュ計算の結果で定期チェックの時刻（のうち分）を決めて，負荷分散を行っている．

6. 端末の自律性確保

パッチシステムと定期自己診断機能は，どちらも端末機自身で動作するプログラムで実現されているため，端末の動作異常の際にはこれら機能自身が働かないおそれがある．たとえば，NIS を用いたシステムは起動時に NIS サービスの障害時にはネットワーク経由の操作がいっさいできなくなる状態に陥ることさえある．

本システムでは，パッチシステムが動作するための最低限の情報（自局とサーバのホスト名，IP アドレス，NFS マウントテーブル，プログラム・ファイル等）はローカルファイルに置くことでネットワーク関連の障害時にもパッチシステムと自己診断機能だけは自律して動作するようにした．

このようにローカルディスクに置く情報を増やすことはトラブルに対する耐性を高める効果がある反面，各端末に分散した設定情報の管理/更新のコストが増

大するおそれがある．本システムでは情報配布の手間はパッチシステムを用いることで問題ないレベルに低減することができる．

6.1 IP アドレスの設定方式

多数の計算機の集中管理を行う場合，IP アドレス等のネットワーク設定は個別には持たせず，bootp, rarp，DHCP 等で集中管理することがよく行われている．また，管理負荷の軽減を目的として，NIS 等のディレクトリサービスを用いてシステム管理情報をネットワーク経由で参照させることが多い．

しかし，我々はローカルファイルに持たせる方式をとった．

これは，以下の 3 つの目標のためである．

- コストを削減し，1 台でも多くの端末を導入する．
- 当初の設置調整の際に，端末設置作業が端末側だけで完結する．
- 故障修理に関してセンター職員の作業負担をなくす．

DHCP 等の利用にはブロードキャストドメイン内に必ずコンフィグレーションサーバが存在するような構成にする必要がある．ルータベースのキャンパスネットワークをまたいで小規模（数台〜）の分散端末室を多く置かなければならない構成では，それぞれのブロードキャストドメインにサーバを置くのは難しかった☆．

また，トラブル追求等管理の都合上，IP アドレスの割当ては固定的に行うことが必要である．このため，DHCP を用いる場合でもサーバには端末の MAC アドレスを登録しなければならない．工場でのキッティングの際に MAC アドレスを調べたうえで 1 個体ごとに設置場所を指定して輸送することは時間的にもコスト的に無理がある．また，設置当初に端末ごとの MAC アドレスを調べてサーバに登録するまでは端末が動き出さないというのでは設置調整の作業効率が悪い．さらに，MAC アドレスを収集するのはそれぞれにキーボードからホスト名を与えるよりも手間がかかる．

また，故障修理によって MAC アドレスが変わったときに MAC アドレス登録変更作業等が必要になってしまうことも問題である．一般には，端末の故障修理の結果 MAC アドレスが変わるかどうかは修理作業を行ってみなければ分からない．MAC アドレスに依存した管理形態をとっていると，サーバへの MAC アドレス登録を更新しない限り修理が完了した端末が動作しない．すなわち，現場でのハードウェア修理のほかにサーバマシンでの特権ユーザ（root）権限を用いた作業を同期して行う必要がある．

サーバに遠隔ログインして MAC アドレス登録が行える修理技術者を想定することも，修理作業時にサーバ室で別の技術者を待機させることも，コスト的には容認し難い．

これらを総合して考えて，IP アドレスはローカルディスク上に保持することにした．この結果，ハードディスク以外の部品は何を交換してもソフトウェア的設定作業は生じない．

ハードディスクの交換後には導入当初の初期設定時と同じくホスト名をキーボードから与えるだけで済む．

ハードウェア保守エンジニアが，端末，サーバともに（運用段階での）特権パスワードを知る必要がないので，大学側もベンダ側も互いに安心である．

6.2 フェイルセーフな起動

NIS や NFS 等でに依存した端末は，ネットワーク系の障害が起こるとシステムがマルチユーザモードに起動しなかったりブート過程でハングアップしたりする．これでは，工場やネットワーク工事中の教室での設定作業や動作確認作業が困難である．また，ネットワークの障害時に端末操作がいっさいできないという問題も起こる．

また，ネットワークトラブル等が生じた際にシングルユーザモードに落ちたり halt したりすると，その端末まで人が行って復旧作業をしなければならない．

そこで，本システムの端末はサーバにアクセスできない場合は自動的にスタンドアロン状態で起動するようにした．端末はブート過程で，まずサーバのパッチシステム情報の部分だけを NFS でマウントすることを試みる．これがタイムアウトや手動操作による割込みで失敗した場合は，NIS 機能を使用しない．そのため，ハングアップせずにマルチユーザモードに移行する．

このスタンドアロン状態は NFS や NIS に依存しないというだけでネットワーク機能は利用可能である．

これにより，工場での組み立て時等にはネットワークに接続しないままコンソールからのログインや GUI アプリケーションの起動テストを行うこともできる．また，サーバの障害時にも管理者だけは普通に遠隔ログインして作業を行うことができる．

何らかの障害によりスタンドアロン状態で起動したあと放置された端末は，5 章で述べる自己診断により，サーバへのアクセスが可能になると自動的に再起動し，一般利用者へのサービスを再開する．

☆ 利用できるネットワークの機能およびネットワーク系に費やすことのできる予算規模によりこの判断は変わりうる．

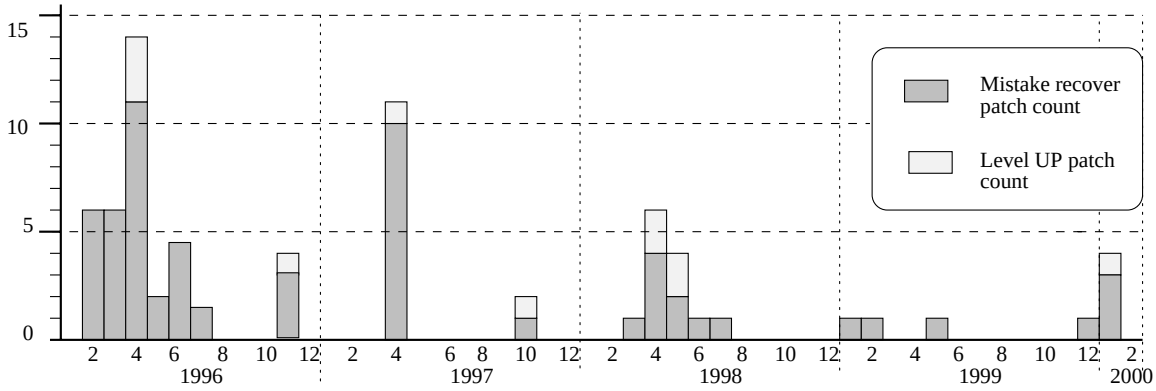

図 3 パッチシステム利用回数
Fig. 3 Patch system usage.

7. 評価と考察

上記のシステムの運用方式について評価し，考察する．

7.1 運 用 結 果

図 3 は，大阪大学におけるパッチシステムの利用回数の月ごとの記録である．導入当初だけでなく，全利用期間を通じて散発的に設定変更の必要は生じていることが分かる．

導入当初は 2 月 29 日検収，4 月 8 日開館というスケジュールであったため，初期設定を順次行った 2〜4 月の利用回数が増えている．毎年 3〜4 月には，追加ソフトウェアの購入やネットワークの構成変更（VLAN 導入等）や学部/学科改組にともなうファイルサーバの構成変更を行うためパッチシステムの利用が発生している．その他の時期は，ソフトウェアのバージョンアップや発見されたセキュリティホール，y2k への対応等である．

このパッチシステムの運用上の問題は，パッチスクリプトの記述が難しいことである．図 1 の例では，書き換えに失敗すると起動しなくなるファイル（fstab）に関しては，上書きするのではなく別名のファイルを作ってから置き換えている．このような細かな配慮がゆきとどいたスクリプトを作成するのは難しい．実際，図 3 にみるように 4 年間に発行したパッチは 69 個であるが，うち 11 個（薄い色の部分）はパッチ・スクリプトの書き損ね等の復旧のためのパッチである．

本パッチシステムは，同じく NEXTSTEP を採用した大阪市立大学（平成 8 年 10 月運用開始）や高知工科大学（平成 9 年 4 月運用開始）のシステムでも用いられた．大阪市立大学のシステムでは，3 年 9 カ月で 51 件のパッチを発行しており，大阪大学とほぼ同じ頻度である．

7.2 評　　価

パッチスクリプトで行っているような設定変更は，リモートシェルを用いてサーバから端末を操作しても行える．しかし，サーバでの作業時に停電や故障で停止している端末は，記録したうえで次に正常稼働するまで待たなければならない．また，端末の再起動を要する変更作業が生じたときには，閉館時間（情報処理教育センターでは 21:50）まで待つ，休日に出勤するあるいはサービス停止のアナウンスを行ったうえで端末室を一時的に閉室して作業しなければならない．

本システムでは，端末が適切な時刻に自主的に設定をダウンロードするので，端末の状態を気にせず，サーバにパッチを登録できる．再起動を要する設定変更の際も，通常は利用者がログアウトした際に 5 分ほどその端末が利用できなくなるだけで済む．そこで，授業時間中だけを避けて，自習利用時間帯にはいつでもパッチを発行することができた．

意図したとおり，パッチシステムにより，全端末を巡回しての設定変更作業は 4 年間を通じて行う必要がなかった．不具合を発見してから対処が完了するまでの時間は，巡回しての作業では，投入する作業員の人数にもよるが数日を要する．パッチシステムを用いることで，所要時間が数時間（スクリプトの記述とテストに 1〜2 時間，登録してから各端末で適用されるのを待つ時間が 1 時間）に短縮された．

つづいて，本管理方式によって節約できた管理作業量を見積もってみる．

本センターの端末台数は 500 台で，2 つのキャンパスの 20 の教室に分散して設置されている．

各端末の設定変更作業は，単純に端末のコンソールからログインして行ったとすると再起動の待ち時間を

入れて 15〜20 分程度を要する．全端末を巡回して作業するのに，延べ 125 人・時間という計算になる．実際にはこれに移動時間が必要で，さらに修理中等の理由で一斉巡回時に作業を行えなかった端末は別途後日訪れて作業を行う必要があるので，約 1 人月の作業量と考えられる．

パッチシステムを用いると，故障している端末を除き設定変更は 1 時間程度で完了するが，作業員が巡回して作業を行うとすると，設定変更の要求が生じてから完了まで 2〜3 日の猶予を置いたとしても，設定変更 1 回あたり 5 人程度の作業要員が節約できたことになる．ただし，パッチを記述するためには，高技能のスタッフが 1 時間程度作業しなければならない．

なお，本パッチシステムでは利用者のログアウトを待ってパッチ適用を行うことで，作業によるサービス停止を避けることができる．手動作業による設定変更作業で，使用中の端末は利用者がログアウトするまで待ったとすると膨大な待ち時間が生じ，作業人員の必要数が数倍になると考えられる．

パッチシステムにより，初期導入時の設定作業は 1 台あたり 2〜3 分で完了する．本来の GUI の設定ツールで個別設定を行ったとすれば，1 台あたり OS 導入に 1 時間☆，その後の個別設定に約 30 分を要したと考えられる．

また，設定作業が非常に簡単になり，OS（NEXTSTEP）に精通している必要がない点も重要である．そのため，端末の設置とケーブリングの担当エンジニアに，ついでに設定を依頼することができた．

単純計算で，OS の導入・カスタマイズが可能なエンジニアの作業量が 200〜600 人・時間☆☆節約できたことになる．また，ネットワーク工事が完了していなくても設定作業が行えた（後日，サーバまでのネットワークが開通した時点でパッチシステムが働く）点も作業日程の短縮に貢献した．

導入作業の容易化は，システム稼働後の保守作業の効率化にも役立った．ハードディスクの故障等で OS の再インストールを必要とする端末は月に数台生じる．本システムでは OS の再設定の際の作業は実質的にはキーボードからのホスト名入力だけなので，ハードディスクの交換を行った修理要員に任せることができた．パッチシステムがなければ，これは本来別途 SE かセンター職員が 30 分〜1 時間ほどかけて行わなければならない．また，端末の前まで行かなければならないので，往復 1 時間（豊中地区から吹田地区に設定に赴く場合）の移動時間も必要である．

自己診断機能により，端末の設置場所を訪れて再起動を行う必要性が低下した．4 年間の運用期間を通じて自己診断の結果の再起動は各 200 回程度起こっているが，工事停電による通信の途絶等単一の原因で（1〜6 時間ごとに）複数回再起動した事例を考慮すると，実質的には 1 カ月に 1 回強の頻度である．これは，全体の台数を乗ずると，毎日 20 台程度を再起動する手間が省けたことを意味する．端末の地理的分散も考慮すると，これは再起動専業の要員を 1 人置くことが必要な作業量である．

7.3 一般性に関する考察

本方式の特長は，各端末の設定情報を設定手続き（スクリプト ＋ データファイル）の形で保管すること，各端末が自律的に設定の更新や自己診断を行うこと等にある．個別設定ファイルをあえて端末に配布することで自律性を向上させている．個別端末上で設定ファイルを維持するための管理の手間の問題は，更新の自動化によって解決している．

Unix および類似・近縁の OS（BSD や linux 等）では，システム管理・設定は GUI だけでなくコマンドラインからも行えるのが通例なので本方式との親和性は良い．また，定時実行機能（cron）から特権で動作する監視・管理プログラムを起動することも容易である．ただし，書籍やマニュアル等で解説されている管理作業は GUI や対話的コマンドでのものなので，これを非対話的スクリプト経由の作業に翻訳する負荷が運用担当者にかかってくる．

Windows では，パッチシステムの基本アイディアは適用可能であるが，設定手続きをシェルではなく，バッチスクリプト，VBA（Visual Basic for Applications），WSH（Windows Scripting Host），perl/win32 等で記述することになろう．また，GUI によるソフトウェアのインストールや設定変更を，変更されるファイルおよびレジストリ項目情報の形に変換する手間がかかる．ただし，レジストリの変更項目を検出・把握することは大規模集中管理環境では通例行われることなので，パッチシステムの利用のための追加の作業が膨大になることはないと考えられる．

その他の OS の場合，管理作業をスクリプト言語の形で記述する機能を持つものならば，本方式は適用可能と考えられる．

続いて，有効性について考察する．

☆ CD-ROM 等インストールメディアの高速化は著しいが，ハードディスクの容量の増加も急激であるので他のシステムに適用した場合でもこの所要時間は大きくは変わらないと考えられる．

☆☆ 一斉導入の対象となったのはレンタルの端末で，500 台中 400 台である．

本管理方式の中核は設定の配布・復元を自動化するパッチシステムである．

本方式は全端末でほぼ同じ環境を提供する，コンピュータセンター的端末群の管理に適しているといえる．管理対象端末の設定が個別に大きく異なる場合にはパッチスクリプトの記述が困難となる．また，部門ごとの独自管理が行われている場や各利用者個人が管理者特権を持つ個人常用 PC の場合は，パッチシステムによる中央集権的管理とローカル管理が干渉するので不適と考えられる．

また，本センターのように端末の設置場所が地理的に分散しており管理スタッフが少数でスタッフが常駐しない（必要に応じて訪問する）設置場所が多い場合に有効性が高い．

本方式ではパッチスクリプトの記述のためにとりわけスタッフの熟練度を必要とする．よって，全端末を訪問して作業を繰り返すことが許容可能な場合，すなわち，端末台数が比較的少なく管理者の常駐場所の近隣（たとえば隣室）に集中して設置されている場合には本管理方式の導入による作業量の節約は大きくはないと考えられる．

ただし，教室利用時間の都合で平日昼間に閉館できず，設定変更の巡回を深夜勤務や休日出勤で行わなければならないようなシステムでは，より少ない台数でも本方式が有用となる．

本方式では，全端末での設定の変更やファイル配布は，作業時間としてはパッチスクリプト開発，待ち時間は各端末のパッチシステム実行間隔（本学では 1 時間）で完了する．よって，この時間と全端末を巡回して設定を行った場合の所要時間の差が，本方式の採用で得られる省力化量と見積もることができる．

7.4 他の管理方式・提案との比較

システム管理を省力化する手法やツールは多くのものが提案されている．

NIS，NDS，LDAP 等はユーザアカウント等の情報を集中管理するものである．教育用計算機システムでは利用者や管理者の階層的権限関係や階層的権限委譲はないフラットな空間で十分なので LDAP のような高度な管理機能は必ずしも必要ではない．

既存の管理システムと称するものの多くは，以下のような機能を持つことが多い．

- 個々の機器のハードウェア構成・ソフトウェア構成を検知する．
- ネットワークの構成を自動検知する．
- 機器の状況を監視し，障害の発生を検知する．
- 障害の解析を補助する．

また，文献 4) のトラブルチケットシステム等は，運用管理にかかわる事務作業を計算機で支援するものである．

以上の管理方式・システムは本稿の管理手法とは観点が異なるもので，省力化の対象が異なる．また，本方式の運用と並行して利用可能である．

ただし上記のうち障害検知システムは，MTTR 短縮等管理の質の向上には役立つが，実際の修復作業の量を減らすものではない．

端末の集中的な構成管理は，設定ファイルの自動コピー機能を用いることが多い．たとえばサーバ/クライアントともに OS が UNIX である場合には，ファイルシステムの内容の同期させる rdist や rsync が用いられる．これらは，サーバに格納した設定ファイルの雛型と各被管理ホストの設定ファイル群を比べ，異なるファイルを転送する．

ファイルシステム同期化ツールは，ひとたび設定すると更新されたファイルを自動的に検出して転送するので運用が楽である．しかし，被管理ホストの構成が複数種類ある場合にはその種類ごとに雛型を用意しなければならない．また，ファイル比較の負荷も問題である．上書き更新を行うことが不適当なファイルが存在するという問題もある．

さらに，近年の OS によく取り入れられている設定データベース☆の問題がある．これらは，多種多様な設定を 1 つのデータベースファイルに格納している．設定の特定項目だけを変更/カスタマイズするためには，ファイルコピーでは不可能で，データベースの特定レコードを変更する操作を行う必要がある．

また，大規模システムでは更新作業実行時に停止している端末の扱いが問題になる．この点でも，3 章で論じたように端末側が能動的に動作するプル型が優れている．

分散配置した端末の設定内容更新手法としては，ネットワークワームを用いた手法[5] も提案されている．ワームを用いた場合，ワームの生存時間の間停止し続けた端末が存在した場合対処できない．一方，ネットワークトラブルの場合本方式は回復するまで待つだけだが，ワームの場合は迂回路があれば設定更新情報を行うことができる．ただし，本センターのようなシステムではセンターサーバとの直接通信ができない状況では利用者へのサービスもいっさい行えないので，そのような状況で設定の変更だけができても得るところは少ないと考えられる．

☆ NEXTSTEP では netinfo.

8. ま と め

本稿では，多人数が利用する，多数の計算機をネットワークで接続した分散型システムの維持・管理を省力化するための方法を提案した．

設定内容をスクリプトの形で記述したことと，各非管理端末が自律的に動作する方式をとったことで，設定内容の更新および復元の手間を大きく省くことができた．

謝辞 有益な意見をいただいた情報処理教育センターの教職員および学生ボランティアスタッフの皆様に感謝します．

参 考 文 献

1) 齊藤明紀：大阪大学情報処理教育センターのシステム構築 (1)〜(6)，*UNIX MAGAZINE*, 1996年 5, 8, 12 月号，1997 年 1, 2, 3 月号，アスキー．
2) 齊藤明紀ほか：教育用計算機システムの運用および授業支援のためのシステムの現状と今後，情報処理学会分散システム運用技術研究グループ資料，DSM-9501031 (Jan. 1995)．
3) 齊藤明紀ほか：多人数教育用計算機環境におけるシステム管理の省力化の一方法，情報処理学会分散システム運用技術研究会 (Jnly 1997)．
4) 宇多ほか：ネットワークテストベッドにおけるトラブルチケットシステムの構築と運用，情報処理学会分散システム運用技術研究会報告，DSM9-1 (May 1998)．
5) 小野木ほか：ネットワークワームを利用したネットワーク管理手法，コンピュータソフトウェア，Vol.6, No.3 (1999)．

(平成 12 年 5 月 9 日受付)
(平成 12 年 10 月 6 日採録)

齊藤 明紀（正会員）

平成 3 年大阪大学大学院博士課程修了．同年同大学基礎工学部助手．平成 6 年大阪大学情報処理教育センター講師．現在，大阪大学大学院基礎工学研究科講師．工学博士．分散システム運用技術，教育工学，ユーザインタフェース等の研究に従事．電子情報通信学会会員．