The IDA

Open Source

Migration Guidelines

netproject Ltd
124 Middleton Road
Morden, Surrey
SM4 6RW

Tel: +44 (0)208 715 0072
Fax: +44 (0)208 715 7134
Web: www.netproject.com

October 15, 2003
Ref: OSPL/EEC-01.10

0.1. Document History

Date	*Version*	*Author*	*Changes*
11/02/03	0.1	S. Hnizdur	Initial draft
13/04/03	0.2	S. Hnizdur	Additions and re-organisation
14/05/03	0.3	S. Hnizdur	Additions and re-organisation
20/05/03	0.4	S. Hnizdur	Preparation of draft for PMB meeting
21/05/03	0.5	S. Hnizdur	Inclusion of V0.4 comments
31/05/03	0.6	S. Hnizdur	Inclusion of V0.5 comments
30/06/03	0.7	S. Hnizdur	Inclusion of V0.6 comments and MS server apps
16/07/03	0.8	S. Hnizdur	Inclusion of V0.7 comments and Unix Scenario.
17/07/03	0.9	S. Hnizdur	Inclusion of V0.8 comments and Windows desktop apps
10/08/03	0.91	S. Hnizdur	Inclusion of V0.9 comments
27/08/03	0.92	C. P. Briscoe-Smith	Corrections and inclusion of some V0.91 comments
04/09/03	0.93	C. P. Briscoe-Smith	Corrections
08/09/03	0.94	C. P. Briscoe-Smith	Further corrections and inclusion of comments
14/09/03	0.95	S. Hnizdur	Re-organisation after PMB meeting.
25/09/03	0.96	S Hnizdur	Corrections.
07/10/03	0.97	S.Hnizdur	Final Draft
10/10/03	1.0	S.Hnizdur	Version 1 issue

0.2. Distribution

netproject Ltd	x1
Frequentous Consultants Ltd	x1
European Commission (for dissemination to member Administrations)	x2

0.3. Trademarked terms

Trademarks have been used in this document for purposes of identification only. The authors acknowledge the ownership of these trademarks.

0.4. Copyright

0.5. Table of Contents

1. Preface....8
1.1. Abbreviations and terminology....8
1.2. Audience....8
1.3. Authors....8
1.4. Acknowledgements....8

PART 1 Introduction and Summary

2. Introduction....11
3. Summary....12
4. Methodology....14

PART 2 Management Guidelines

5. Migration Overview....17
6. Human Issues....21
7. Making Life Easy....23
7.1. Introduce new applications in a familiar environment....23
7.2. Do the easy things first....23
7.3. Think Ahead....23

PART 3 Technical Guidelines

8. Reference Architecture....26
8.1. Generic architectures....26
8.2. Base Reference Architecture....28
9. Functional Groups....30
9.1. The Principal Groups....30
9.1.1. Office....30
9.1.2. Mail....30
9.1.3. Calendaring and Groupware....30
9.1.4. Web Access and Services....30
9.1.5. Document Management....30
9.1.6. Database....30
9.2. The Subsidiary Groups....30
9.3. General Considerations....31
10. The Reference Model – Summary....32
10.1. The Desktop....32
10.1.1. Office....33
10.1.2. Mail....33
10.1.3. Calendaring and Groupware....33
10.1.4. Web Access....33
10.1.5. Document Management....33
10.1.6. Databases....33
10.2. The Servers....33
10.2.1. Mail....34
10.2.2. Calendaring and Groupware....34
10.2.3. Web Services....34
10.2.4. Document Management....34
10.2.5. Databases....34
11. Applications – Principal Groups....35
11.1. Office....35
11.1.1. OpenOffice.org and StarOffice....35
11.1.2. Koffice....36
11.1.3. Gnome Office....36
11.2. Mail....37
11.2.1. MTA....37
11.2.2. Mailstore....38

11.2.3. MUA....38
11.2.4. Anti-Virus....39
11.2.5. Other Tools....39
11.2.6. Problems Experienced....40
11.3. Calendaring and Groupware....40
11.3.1. Personal calendars and agendas....42
11.3.2. Group calendars....42
11.3.3. Meeting organisation....42
11.3.4. PDA synchronisation....42
11.4. Web services....42
11.4.1. Browser....42
11.4.2. Web servers....43
11.4.3. Portal / Content....44
11.5. Document management....44
11.5.1. Registration and retrieval....44
11.5.2. Collaborative working....44
11.6. Databases....45
11.6.1. Central databases – application-based....45
11.6.2. Personal databases held centrally or locally....45
11.6.3. Database Connectivity....45
11.6.4. Performance....45
12. Applications - Subsidiary Groups....47
12.1. Operating System....47
12.2. User Interface....48
12.2.1. Desktop manager – look and feel....48
12.2.2. Language....48
12.3. Security....48
12.3.1. Data encryption....48
Data in transit....48
Data in storage....48
12.3.2. Authentication....49
12.3.3. Authorisation....49
12.3.4. Virus control....49
12.3.5. Proxy Server....49
12.3.6. Firewalls....49
12.3.7. Virtual Private Networks (VPN)....50
OpenVPN....50
FreeSWAN....50
CIPE....50
12.4. Management....50
12.4.1. User management....50
12.4.2. Configuration management....50
Manual Configuration Maintenance....51
Cfengine....51
System Configurator....51
12.4.3. Software management....51
System Installation....51
Software Maintenance....53
12.4.4. Hardware management and system monitoring....54
MRTG and Snmpd....54
Nagios....55
smartd....55
12.4.5. Printer management....55
LPRng....55
Common Unix Printing System....55
Kprint and GnomePrint....55
12.5. Backup and recovery....55
12.5.1. Dump and Restore....56

12.5.2. Amanda....56
12.6. Other services....56
12.6.1. Time Servers....56
12.6.2. Network infrastructure servers....56
Routing....56
DNS....56
DHCP....57
12.6.3. File servers....57
NFS....57
Samba....57
Netatalk....57
OpenAFS, CODA and Intermezzo....57
12.6.4. Directory services....58
12.6.5. Legacy support....58
Terminal emulation....58
Remote display....58
Emulation....58
13. Application Migration - Overview....59
13.1. Proprietary applications which have an OSS equivalent....59
13.2. Proprietary applications which run in an OSS environment....59
13.3. Software which may be accessed by remote display....59
13.4. Software which will run under an emulator....60
13.4.1. Hardware emulation....60
13.4.2. Software emulation....61
13.5. Software which can be recompiled under OSS....61
14. Scenario 1 – Windows....63
14.1. Planning the migration....63
14.2. Domains....63
14.2.1. Windows “workgroup” model....63
14.2.2. Windows NT domain....63
14.2.3. Windows 2000 Active Directory domain....64
14.3. Overview of possible migration routes....64
14.4. General issues....66
14.4.1. Usernames and passwords....66
Username issues....66
Password issues....67
14.4.2. Authentication services....67
14.4.3. Files....67
Content and format....67
File names....67
Dual access....68
14.5. Tools....69
14.5.1. Samba....69
14.5.2. OpenLDAP....70
14.5.3. NSS and PAM....70
14.5.4. GNU/Linux SMBFS file access....70
14.5.5. Winbind....71
14.6. Migrating the operating system environment....71
14.6.1. Add individual GNU/Linux servers to an existing Windows NT domain....71
14.6.2. Run GNU/Linux desktops in Windows NT domains....71
Simple setup for small numbers of machines....71
Smarter setup for larger rollouts....73
14.6.3. Run GNU/Linux desktops in Active Directory domains....74
14.6.4. Replace Windows NT PDC/BDC with Samba+LDAP....75
14.6.5. Replace Windows 2000 Active Directory with LDAP....76
14.6.6. Run parallel GNU/Linux infrastructure and migrate users in groups....76
Replacing all Windows clients with GNU/Linux....76
Keeping some Windows clients....77

14.7. Migrating server-side applications....77
14.7.1. Web servers: moving from IIS to Apache....77
Migration issues....77
Migrating a static website....79
A simple WebDAV configuration....80
14.7.2. Databases: moving from Access and SQL Server to MySQL or PostgreSQL....81
Migrating Access Databases....81
Migrating SQL Server databases....82
Database migration issues....83
14.7.3. Groupware: moving away from Exchange....83
General issues....83
Mail issues....84
Addressbook issues....84
Calendar issues....84
14.8. Migrating desktop applications to OSS....84
14.8.1. Office....84
Document conversion....85
Template conversion....85
Macro conversion....85
Word Processing....85
Publishing....86
Spreadsheets....86
Presentation Graphics....86
Graphics and image manipulation....87
PDF generation....87
14.8.2. Mail....88
14.8.3. Calendars and Groupware....88
Calendars....88
Contact management....89
14.8.4. Web Browsing....89
14.8.5. Personal Databases....90
14.9. Migrating print services to OSS....90
14.9.1. The Windows print model....90
14.9.2. The Unix and GNU/Linux print model....91
14.9.3. Setting up an OSS-based printing service....91
14.9.4. Printing from Windows clients to GNU/Linux-attached printers....92
Using the lpr protocol....92
Using printer shares....92
Using Point and Print configuration....93
14.9.5. Printing migration schemes....93
14.9.6. Potential Problems....93
14.9.7. Further information on printing....94
14.10. Legacy applications....94
14.11. Virus protection....94
14.12. References....95
15. Scenario 2 – Unix....96
16. Scenario 3 – Mainframe....98
17. Scenario 4 – Thin client....99

Appendices

Appendix A....101
Appendix B....103
Appendix C....107
Appendix D....114
Appendix E....122
Appendix F....136
Appendix G....141

1. Preface

1.1. Abbreviations and terminology

Wherever possible, the first time an abbreviation is used the expanded version will also be included. A glossary of terms is provided in Appendix G. Where a glossary term is first introduced it will be presented in this style: ***Glossary***.

The terms ***Open Source Software*** and ***Free Software*** each have their champions. In this report we use the term Open Source Software or ***OSS*** and intend this to mean that the software described has the characteristics implicit in both the terms Open Source Software and Free Software. For more on these terms see http://www.gnu.org/philosophy/categories.html/ and http://www.opensource.org/.

Product names will be presented in this style: `Product Name`.

Operating system terms such as file names will be presented in this style: **Filename**.

Program code will be presented in this style: `Code`.

1.2. Audience

This document is intended for IT managers in public administrations within Europe. The term ***Administration*** is used throughout the guidelines to mean a European public administration and the term ***Administrators*** to mean this group of people.

1.3. Authors

The report has been produced by **netproject Ltd** in conjunction with **Frequentous Consultants Ltd.** The content was produced by a number of consultants including:

- Steve Hnizdur
- Keith Matthews
- Eddie Bleasdale
- Alain Williams
- Andrew Findlay
- Sean Atkinson
- Charles Briscoe-Smith

1.4. Acknowledgements

netproject would like to thank the following for their help:

- In particular **netproject** would like to thank the many members of the Project Management Board for the lively and informative discussions every month.
- Thomas Krupp and Heiko Thede (Mecklenburg-Vorpommern)
- Frank Müller (Landesrechnungshof Mecklenburg-Vorpommern)
- Mru Patel (Sun Microsystems)
- Graham Taylor (Open Forum Europe)

- Richard Heggs (Nottingham City Council)
- Andy Lack (City University London)
- David Amos (BSS Group)
- Steven Pennant (London Borough of Newham)
- Andy Trevor (Total Solutions)
- Geoff Gunton (Northern Rock)

PART 1
Introduction and Summary

2. Introduction

The purpose of these guidelines is twofold:

1. To help Administrators decide whether a migration to OSS should be undertaken.
2. To describe in broad technical terms how such a migration could be carried out.

The guidelines are intended to be of practical use to Administrators and so must be relevant and accurate whilst being accessible and understandable. They are not a detailed technical reference manual. The structure is intended to allow changes to be easily made in the future as experience is gained by Administrations and as changes are made to the products available.

To meet these objectives it is imperative that the content is kept up to date and that any inaccuracies are removed. To this end, comments and contributions are encouraged from readers on any part of the guidelines. Please send comments to gposs@cec.eu.int**.**

The guidelines do not talk about OSS in general or about the relative merits of the various licenses. This, together with a great deal of other information, can be found in the following IDA documents:

1. The OSS Fact Sheet:

   http://europa.eu.int/ISPO/ida/export/files/en/840.pdf

2. The Report on OSS usage:

   http://europa.eu.int/ISPO/ida/export/files/en/837.pdf

3. The Report on market structure and issues related to public procurement:

   http://europa.eu.int/ISPO/ida/export/files/en/835.pdf

(All three of the above documents can be found at:

http://europa.eu.int/ISPO/ida/jsps/index.jsp?fuseAction=showDocument&parent=crossreference&documentID=333/

in other formats as well as PDF.)

The French “Agence pour le Développement de l'Administration Électronique”, ADAE, have produced a good guide on licences, which is also available in English at:

http://www.atica.gouv.fr/pages/documents/fiche.php?id=1450&id_chapitre=8&id_theme=55&letype=0.

3. Summary

These guidelines are for IT managers and practitioners who are planning or doing a migration to Open Source Software (OSS). They are based on the practical experience of the authors and the content of a limited number of publicly available case studies. They have been validated against the migration to OSS in the Court of Auditors, Schwerin in Mecklenberg Vorpommern.

There are many reasons for Administrations to migrate to OSS. These include: the need for open standards for e-Government; the level of security that OSS provides; the elimination of forced change; the cost of OSS. All these benefits result in far lower IT costs.

The guidelines recommend:

- before starting have a clear understanding of the reasons to migrate;
- ensure that there is active support for the change from IT staff and users;
- make sure that there is a champion for change – the higher up in the organisation the better;
- build up expertise and relationships with the OSS movement;
- start with non critical systems;
- ensure that each step in the migration is manageable.

Migrating IT systems provides an opportunity to re-engineer them to meet the new demands placed on them. Questions that need to be addressed include:

- how to ensure the interoperability of systems;
- how to support mobile users;
- how to securely identify remote users;
- how to build systems that are manageable.

Above all how to ensure that security is designed in from the start and not tacked on as an after thought.

For server computing, OSS is well understood and is extensively deployed. Migrating servers to OSS can generally be done without having any adverse effects on the users. It is normally the first place to start.

Deploying OSS on the desktop for most organisations offers the largest cost savings. When migrating the desktop the new OSS applications will have to inter-work with existing applications. In particular, the way in which group calendering interworks in both OSS and proprietary desktops must be addressed.

When replacing proprietary office automation software templates must be checked to ensure that they produce the correct output. Macros need to be re-written – preferably as scripts. Applications for which there are no OSS equivalents can be run as thin clients. Over time desktop applications can be replaced with OSS equivalents.

Although the guidelines assume a complete change to OSS the likelihood is that a heterogeneous environment will be built particularly as a migration of thousands of desktops will take time. Mixtures of OSS and proprietary applications are also likely because replacement OSS applications may not always be available or suitable. At the moment this is particularly true in the replacement of Microsoft's Exchange groupware function. However there are enough OSS applications of sufficient quality to make migration compelling.

It is important to make sure that decisions made now, even if they do not relate directly to a

migration, should not further tie an Administration to proprietary file formats and protocols.

OSS is a disruptive technology. It enables a fundamental change to the way organisations provide IT services. It is a move away from a product to a service based industry. OSS software costs nothing to install. The issue is where to get support. There are a number of third party support companies as well as the distribution vendors. However if your attitude to IT is “Who do I sue when things go wrong?” then perhaps OSS is not for you. An understanding of the dynamics of the OSS movement is necessary. Knowing how to relate to the OSS community is advisable.

4. Methodology

Any migration exercise should consist in general of:

1. A data gathering and project definition phase including:
   A. A description of the set of relevant initial conditions consisting of, for instance:
      a. systems architecture(s),
      b. applications and their associated data,
      c. protocols and standards used,
      d. hardware,
      e. the physical environment, such as network bandwidth, location,
      f. the social requirements such as language(s) and staff skill sets;
   B. A set of target conditions in the same detail,
   C. A description of how to get from the existing to the planned conditions;
2. A justification for the migration including the cost associated with it,
3. One or more pilot phases which are designed to test the plan and the justification. Data from these pilots can then be fed back into the cost model used in the plan,
4. Roll out of the plan,
5. Monitoring of the actual experience against the plan.

The contents of item 1 above define something which in these guidelines is called a ***Scenario*** and the guidelines describe how to migrate to OSS in such circumstances.

However, for the guidelines to be readable and useful in practice there have to be some simplifying assumptions, otherwise the total number of possible combinations would become unworkable.

We have chosen one of the many different target environments (1B above) and have simplified the description of the initial environments (1A above). The target environment is discussed in Section 8.2. With this standard target environment assumed, a Scenario becomes defined by reference to the simplified initial environment and the migration path from it to the target.

New chapters can be added to this document as necessary in the future. Starting with Chapter 14, each chapter provides a reasonably detailed description of one Scenario, describing how to migrate to the target including discussion of partial migrations. Each chapter should be updated as experience is gained of real migrations.

In addition, there is an associated spreadsheet to help with item 2 above. In this spreadsheet the cost of each Scenario can be compared to the cost of the target together with the cost of migration.

The detail available in the survey of publicly available case studies was very limited. Only a small number of such studies were found (a list is given in Appendix A) which provided any detail at all other than a general press release. This means that most of the guidelines are based on the experience of **netproject** and its consultants and their discussions with people who have done a migration but not published their results.

The huge number of different combinations of initial and final conditions, together with the many different ways of getting from the one to the other, means that it is impossible for any set of guidelines to cover all the possibilities. The guidelines must therefore be considered as indicative of what can be done rather than prescriptive of what should be done. They should be used as a

starting point in the process of migration. They should not be expected to provide an answer in all circumstances.

The migration is assumed to have a target which is a totally OSS environment where possible and sensible, however, there may be many reasons why proprietary systems may need to be kept or used. The possibility of a partial migration is also discussed.

PART 2

Management Guidelines

5. Migration Overview

Much of what needs to be done to migrate from a proprietary environment to OSS is the same for any migration – for example from *Windows NT* to *Windows 2000*. Even in such single-vendor moves it cannot be assumed that file formats, for instance, will be portable, and suitable testing will be needed before any widespread change is made. All migrations must be based on careful planning.

These guidelines are not intended to be a manual on project management and the Administration is assumed to have sufficient skills to be able to manage the migration properly. The description below is intended to highlight the points that are salient to a migration to OSS.

The migration process should ideally consist of the following parts. Some of them can be done in parallel such as 2, 3 and 4.

Note that the information found may indicate that modifications will be required to the current environment before a migration to OSS can be conceived. This is why even administrations that have no immediate plans for migration, but which wish to keep this option available to them, are recommended to require only open multi-vendor standards and to assess their infrastructure against this (see also Chapter 7.3 below "Think Ahead").

1. Create a team with appropriate skills and management backing. It is important that there is management support otherwise there will be resistance to a move away from the norm of proprietary systems. This support will need to be sufficient to allow at least the building of representative pilots, so a basic business case will need to be produced with perhaps a more detailed one later on when more data is available.
2. Understand the target environment, both OSS software and the base architecture (see Chapter 8), together the various options and choices available. This means training existing staff, recruiting, or using consultants. This will require some initial cost and so requires sufficient management support. There is sometimes an expectation that software which is free can be understood and used without cost. **This is not the case**.
3. The migration is an opportunity to review the base architecture as well as the application software. The architecture recommended in Chapter 8 is one based on centralised control and has a number of advantages discussed in Chapter 8. There may be cost implications in making this change which need to taken into account.
4. It is very important that OSS is understood. There are a number of issues which need to be fully considered before making any decisions.
   A. The implications of OSS licences need to be clarified particularly if the Administration could be deemed to be distributing any changes to software. See the documents referred to in The Introduction for details.
   B. Where there are several choices for any one function – for instance there are at least three good quality OSS spreadsheets available – Administrators need to understand the pros and cons of each product.
   C. The differences between the various distributions must be considered. Some distributions are backed by commercial companies who provide support and fixes. Some have distinct characteristics, for instance Gentoo provides a distribution based on ***source code*** allowing the Administration to more easily customise the software to cater for their own specific needs. All these differences need to be assessed before a choice is made.
   D. Administrators must determine what level of support is required. Commercial support can be

obtained from the developers of the application or distribution if they provide it. If not, third parties can provide support because the source code is available and there are many companies providing such support.

This is a distinct difference from the proprietary software market where detailed support can only be provided by those companies who have the privilege of access to the source. This becomes important if the proprietary vendor goes out of business without releasing the source code.

If all else fails most applications have active mailing lists where a question or plea for help will be answered by someone with an interest in the application. The presence of an active mailing list and user community is often one of the criteria in the selection of software components in the first place.

5. Audit the existing systems. This data will not only be needed to do the migration itself but much of it will also be needed to construct a cost of ownership model for a detailed business case. Compile inventories of:

   A. For each application used:

      a. The application name, version number and contact point to answer enquiries.

      b. How many users require access to it.

      c. What operating system is being used. What operating systems the application can be run under, including environments such as *Citrix*.

      d. What other applications are required on both client and server for the application to work.

      e. What hardware is required. In particular does it require any non-standard or cutting edge hardware.

      f. What protocols it uses to communicate with other applications.

      g. What file formats it requires.

      h. What internationalisation and localisation is required. Multiple languages and currencies may be required.

   B. Data requirements. This should be interpreted in the wide sense which includes, for instance, word processing documents and spreadsheets, sound/voice data and visual data as well as the regular databases. In general, anything which is intended to be processed by a computer.

      a. What are the constraints on interfacing with systems or users outside the control of the Administration?

      b. What requirements are there to keep data and be able to process it in future? Is there a repository of existing legacy data which has to be supported? If so, does it require specific applications to process it?

         i. Divide the data into the following categories:

         ii. Data which is not necessary to keep and can be thrown away. Throw it away.

         iii. Data which must be kept and is currently in an open format such as PDF or Postscript, or can be easily translated into one. The cost of translation needs to be assessed.

         iv. Data which must be kept but which is in a proprietary closed format which cannot be easily translated into an open one. This data may need copies of the specific proprietary application to be kept. The cost of these applications will need to be assessed. The required number of copies of the application can be determined by the degree of access

to the data that is needed. For instance, if the data is only rarely accessed then a single copy on a central machine may be sufficient. It may also be necessary to maintain specific hardware to run these applications.

C. Security requirements.

a. What is the current system for assigning user names and passwords? Do user names have a structure and if so what is it? What is the policy for password updates?

b. Are there any systems requiring authentication other than a simple user name and password challenge?

c. What Administrative and Government policies are there relating to the use of computers? For instance, are there restrictions on the use of the Internet and email?

d. Are there security arrangements which require the use of specific hardware or software?

6. Make a detailed business case for migration. This will be based on the data gathered above and will consist of a number of sections including:

A. The cost of the existing environment over a reasonable length of time such as 5 years with assumptions appropriate to the Administration.

B. The cost of alternative environments and the cost of migration to each over the same period.

C. The strengths and weaknesses of the current environment and the various alternatives.

The associated spreadsheet will help to do the cost comparison.

7. Consult with the users. Explain the reasoning behind the migration and the effects it will have on them. Take their concerns seriously and allow them to play with the technology as soon as possible. The sooner the users become involved the better. This may be a legal requirement in some countries but should be done in any case to ease the introduction of what may be a significant change to working practices.

Create a help desk which can answer user concerns. Later when the migration is underlay it can answer problems and become a centre of excellence and good practice. Create an internal intranet site with a “tips and how-to” section, which can be updated by users themselves. It is important that the users feel included and the site will also give the help desk an idea of the sort of problems being faced by users.

8. Assuming the business case has been made, start with small scale pilot projects, preferably in a self contained environment with a small number of users. This will provide, among other things:

A. Data for more refined Cost of Ownership models.

B. User reaction which can be used to ease the introduction of other systems.

C. Validation or modification of the target architecture and the business case.

9. Decide on the speed of the migration process once it gets started. The main options are:

A. **Big bang:** All users change from the old system to the new one on the same day. In practice this is likely to mean scheduling the change over a weekend or national holiday. The advantage is that no dual-access provisions are needed and staff do not find themselves swapping backwards and forwards between systems. Disadvantages include the high risk and the very large resource requirements during the changeover. This migration scheme is only likely to appeal to small Administrations.

In any case if at all possible **AVOID BIG BANG MIGRATION**. Big bang migrations will have so many variables to control that they almost always fail. If they do so it is unlikely to be

because of a failure of OSS but of management.

B. **Phased transition in groups:** Users are moved from the old system to the new one in groups. It is likely that complete functional groups would be moved together, to minimise data-sharing and group working problems. Risks can be contained and resources managed by choosing appropriate group sizes. It may be possible to do a rolling hardware replacement of desktop PCs at the same time, upgrading the machines removed from one group and then installing them in place of the next group's old machines.

C. **User by user transition:** Essentially the same as the group transition option but with a group size of one person. This drip-feed method has a low resource requirement, but is inefficient and unlikely to appeal to large Administrations. It may be an appropriate way to run the pilot projects though.

It is likely that old and new systems will have to run side-by-side for some time. It is important to have a transition strategy enabling old and new systems to work together, so that production activities can proceed adequately during the transition period. The replacement of the last machine may take a very long time (or never happen), so coexistence is likely to be very important.

10. Roll out the migration to the whole Administration. This will involve further training of users and technical staff.

11. Monitor user feedback and fix any problems which arise. Some user requirements may be sufficiently obscure that they cannot be predicted in advance or discovered during pilot projects. Ensure that sufficient resources are available to deal with such requirements after the transition.

At any time it might be found that migration is not feasible. This could be for instance because there are critical applications which will not work satisfactorily under an OSS environment and the cost to rewrite them is too high.

6. Human Issues

These guidelines are not intended to be a Human Resource Management guide, and Administrations will have encountered many of these issues in other areas before. They will have considerable internal skills on how to overcome them in a sympathetic way and so Human Resource staff ought to be involved from an early stage. The intention here is simply to highlight the sort of issues that have arisen in other sites who have migrated to OSS.

It is very important that all staff are consulted and kept up to date with developments. One way of doing this is to create an intranet which can be easily kept current and which can have a section for user feedback.

Access to training is very important. Some sites have allowed users to decide for themselves if they want to attend whereas others have required attendance. The choice will depend on the culture of the Administration and what the training course is about. Manuals and general documentation is usually only in English and this might cause a problem with some staff. Translation to the local language might be considered as a migration cost but there will then be the problem of the continuing translation of updates.

The OSS user interface in particular *Gnome* and *KDE* provides a choice of languages but the translation may not be complete with some menu items and help screens still in English. *Gnome* in particular has good accessibility features for the visually impaired. Also not all applications will have full localisation support. This is all changing rapidly though and the structure to allow the use of a language other than English is in place if the Administration wants to use it.

There are a number of classic reactions to any change to working practices which will need to be planned for:

- **Fear of the Unknown**

  The use of OSS will be completely new to most users and systems staff. The natural fear of the unknown will mean that people will resist OSS because it is new to them.

  There will be users, who are more naturally inquisitive, who may be very happy to try the new and it is these who should be introduced to OSS in the first instance. The experience so far indicates that once people get over their reserve they find that OSS is not significantly different to use than proprietary software and they become quite happy to use it. It is likely therefore that this initial group of users will successfully transfer to OSS enthusiastically. In any case these people are also likely to ones who will provide useful feedback.

  The first group of users could be used in pilot trials and once they have experience they can be used to coax and educate their colleagues. In any case, in the second phase, users who may be more reserved will need to have greater support facilities in the form of help desks, intranets and experienced local users.

  The same process can be used for systems staff but the level of training is likely to be significant if the existing proprietary environment is not like UNIX. The systems staff in particular need to have their fears allayed at an early stage. They will be a focal point for all the problems that are bound to occur and if they do not believe in the project they will not be able to encourage users in a positive way.

- **The CV dilution effect**

Both systems staff and users may feel that not using the “industry standard” software will impair their ability to develop their career. This is a tricky problem which needs careful management. The Administration will not want to appear heavy handed in its approach but until OSS is widely used then Administrations may encounter this problem quite often.

- **Knowledge is power**

  The people who know the existing systems and setup have a certain amount of power and they may be very reluctant to give this up if the OSS environment is very different from the existing one. This is again requires careful management as these people do have a critical role in running the existing systems. They may need to be among the first to be trained in the new systems so that their position in the organisation is maintained.

7. Making Life Easy

There are a number of considerations which can ease the introduction of OSS.

7.1. Introduce new applications in a familiar environment

Many of the OSS applications will work on proprietary operating systems and this gives the opportunity to introduce these applications without having to totally change the environment. For instance *OpenOffice.org*, *Mozilla* and *Apache* will work under *Windows* and so can be used as a replacement for *Office*, *Internet Explorer* and *IIS* respectively. Apart from being less disruptive, this approach means that user reaction can be gauged on a small scale and plans for user training can be made based on real experience. In addition, problems such as converting file formats, macros and templates can be eased if the old application is kept available for a little while.

This approach does mean that the choice of application in the final target environment will be limited to those which work in the current one. For instance the target browser may be *Galeon* but *Mozilla* is the only one which will run on both *Windows* and GNU/Linux.

7.2. Do the easy things first

Make changes in the first instance which will not disrupt the user population. This means making changes on the server first. These will then provide a platform for the introduction of client changes later. Many of the server side changes will be compatible with the current environment as well, so the disruptive effect will be minimised.

For instance, DNS name servers, DHCP servers and back-end database servers with proprietary databases such as Oracle could all be candidates to be replaced with an OSS equivalent and still interwork with the rest of the current systems as before. The detail of this is discussed later.

There are applications such as *Samba*, which would not be used in a pure OSS environment, but which allow the co-existence of the old proprietary environment and OSS. The early use of these can be very effective in partitioning the environment into manageable chunks.

7.3. Think Ahead

Do as little as possible now that will make future migration more difficult. For instance:

1. Insist that all web development done either in-house or by contractors produces content that can be viewed on all current web browsers, in particular OSS browsers. This should be good practice in any case as Administrations shouldn't require specific software to view their content. Tools such as *weblint* are available to assist in checking the compatibility of web pages.
2. Discourage the indiscriminate use of macros and scripts in documents and spreadsheets; find other ways of providing the functionality. This again should be good practice since the use of these is a common way in which viruses infect systems. Also, macros can be easily used to steal data and to subvert documents, for instance they could make the document say different things depending on who is viewing it and another thing if printed.
3. Insist on the use of open standard file formats, for instance Postscript and PDF.

There is some debate over whether Postscript and PDF are open standards or not. This is more of a debate over strict definitions and in particular who controls the standard. In reality these are the only standard file formats which are widely used at the moment, have publicly available definitions and can be used without significant restriction.

Attempts are being made to create truly open standard file formats based on XML and the *OpenOffice.org* one is a contender. However just because a file is XML based it doesn't mean it will be open.

In particular do not use proprietary file formats for files which are only intended to be read, and not edited, by the recipient. Again this should be good practice because such files are a common way of spreading viruses. Using such formats means that the Administration will be locked into the vendor for a considerable time. Also these proprietary formats can include considerable quantities of ***metadata***, including in particular, previously deleted text, which if viewed by others, could be embarrassing to the Administration. Viewing this metadata is not difficult.

4. When writing documents in collaboration with others use the lowest common denominator format. For instance make use of *Word 97* format rather than the *Word 2000* one. This will increase the likelihood that OSS applications can participate.
5. Make use of open standard protocols. Open standard protocols are defined as those which are free from patents and with an OSS implementation. There are various national sets of standards such as E-gif in the UK, OSOSS in The Netherlands and SAGA in Germany. The focus and content of these frameworks is slightly different but in general they should be adhered to.
6. Develop systems which are based on at least a 3-tier model (see Section 8.1) where the application code is independent of the human interface and data access methods. For instance, if possible, have a browser interface which can be used on an OSS browser. Building applications in this modular way will make it easier to migrate bit by bit. This will not only reduce the scale of any phase of migration but will also reduce the risk of failure. Traditional monolithic client applications are notoriously difficult to handle.
7. Insist that all new applications are written to be portable. This includes using portable standardised languages such as ANSI C, Java, Python and Perl, and using only cross-platform libraries and GUI toolkits such as *wxWindows* (http://www.wxwindows.org/) and the *FOX toolkit* (http://www.fox-toolkit.org/). Avoid architecture specific languages and APIs. Avoid building applications which require the presence of other proprietary applications.
8. Move users away from proprietary mail readers which use proprietary mailstore formats and communicate with servers using proprietary protocols. Most mail applications will store mail using IMAP. If possible find a way of storing address book and calendar information in an open format.

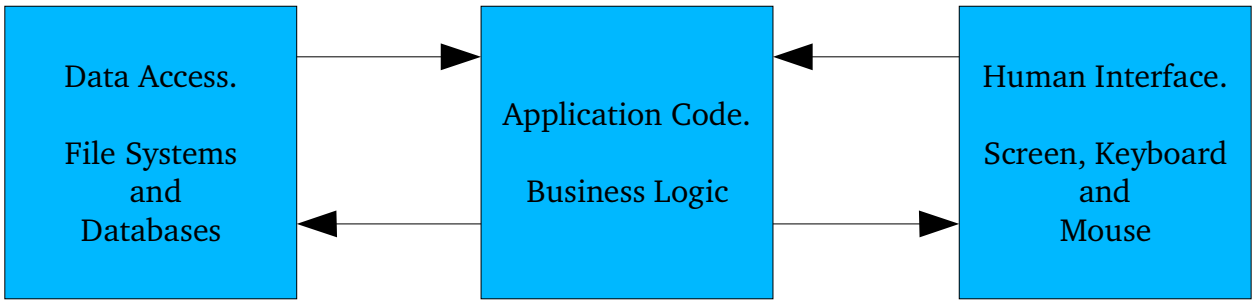
PART 3

Technical Guidelines

8. Reference Architecture

8.1. Generic architectures

One useful way of describing a computer architecture is in terms of the so-called 3-tier model. This model separates out three gross functions that an application performs in general when being used by a human (i.e. it is not a pure server or batch application). These are shown in the following diagram. (http://www.corba.ch/e/3tier.html/ has more details.)



The arrows indicate information being passed between the three parts. These flows should ideally use open and well defined standards. Doing so means that any application only has to worry about performing its business logic leaving the other two functions to standard components. This has the benefit that application code can be simpler and can be more easily run in different environments because its dependence on machine specific access is reduced.

This 3-tier model has been generalised to n-tier, where the components are even further refined, and is typically realised used object or component technology.

Many client-server applications have in the past unfortunately only used a 2-tier model where the application code and the human interface are merged together. This means that migrating such applications is often considerably more difficult than with 3-tier ones. This is because the human interface is likely to require change and the 2-tier application will most likely have human interface code intermixed with business logic throughout.

The communication between the three parts of a 3-tier application normally uses protocols which allow each part, if required, to run on a different machine from the other two. Sometimes the parts can also be split across machines. The choice of location of each of these parts gives rise to various generic architectures.

The extremes from the point of view of the desktop, where at least some portion of the human interface code must be running, are:

1. Ultra thin client

This is where the desktop has only human interface code. Typically it has no local long term memory such as hard or floppy disks. The application code and data access are both run remotely. Examples would be an X terminal, a VT100 green screen and a device with an embedded browser.

2. Ultra fat client

This is where all the code and data is held on the desktop with no network connectivity.

Terms such as thin and fat client mean a desktop which lies in the spectrum between these two extremes.

A variant on these architectures is one in which application code is stored on a server and then downloaded to the desktop to be run when needed. This is the way, for instance, in which Java applets work. Another method involves storing the application code on a server and having the desktop access it as though it were stored locally. This requires the use of a networked file system such as NFS and also means that all the desktops must be of the same processor architecture.

The choice of architecture for any particular application will depend upon:

1. **The network bandwidth to the servers and what that bandwidth will have to carry.** If the desktop is not ultra fat then the network will have to carry either human interface controls, data or downloaded application code. In some circumstances the size of these loads generated by either a single desktop or a number of them in aggregate may be too great for the capacity of the network.
2. **The latency that is acceptable in the use of the application.** When a human interacts with the desktop by depressing keys or moving the mouse the time taken for the application to react and draw the effect on the screen is known as latency. For some applications such as simple data entry high latencies may be acceptable but for highly interactive applications such as drawing, latencies must be low. Latency will depend on the capacity of any network segments between the human interface and the application, and on the capacity of the machine running the application code. To achieve the lowest latencies, therefore, the application should be run on the same machine as the human interface, and this machine should be powerful enough to run the application properly.
3. **The security policy in place**. If data resides on desktop machines distributed throughout the Administration, this means that if any machine is stolen or is accessible in an insecure environment then data can be lost or divulged to third parties or unauthorised staff. This may not be a problem with low grade data which has been properly backed up, but otherwise it could contravene the security policy of the Administration concerning who can access data. Alternatively having data transmitted over a network unencrypted could cause the same problem.
4. **The backup policy in place**. If data resides on desktop machines distributed throughout the Administration then either some centralised backup mechanism is needed or the responsibility for backup has to be distributed among many people, probably the users themselves. A centralised backup scheme would be complex and would require high network bandwidth and co-operation with the desktop users (who, for instance, must remember not to switch off their machine during periods when backups are scheduled).
5. **The design of the application**. If the application has human interface code included then it either needs to run on the desktop or on a server with the human interface code split between the server and the desktop. For instance, an IBM 3270 or DEC VT100 terminal has all the display code in the desktop, as does a browser-based terminal. *Citrix*, *Windows Terminal Server* and the *X Window System* all split the display code between the server and the client.
6. **The capacity of the desktop machine to run the code**. The more the desktop machine has to do, the more powerful (and thus expensive) it has to be.
7. **The capacity of the desktop to store the data**. Some applications need access to huge data stores which can only be held on specialised servers.
8. **The performance of the servers available**. If the application is run on a server rather than the desktop then the server has to be sufficiently powerful to run all the instances of the

application required when the maximum number of desktops are in use. This could mean that the servers have to be vastly over specified to handle the worst case conditions. In addition, the number of desktops which can be supported by a given number of servers tends to have discrete “jumps” in it. This means that the addition of a few extra desktops could mean the purchase of an extra-large server.

9. **The total cost of implementation**.

As with any engineering problem there is no solution applicable to all situations and a particular physical desktop may operate in one way for one application and in another for a different application.

8.2. Base Reference Architecture

The Base Reference Architecture (BRA) used in these guidelines is chosen to be relevant in the majority of situations. It can be extended to be thinner or fatter for specific applications as necessary.

In reality the architecture used by an Administration is likely to be a combination of various architectures each chosen for specific applications.

The BRA can be characterised as a “stateless desktop” in that:

1. All applications run on the desktop whenever possible and are stored on the desktop.
2. No persistent data is held on the desktop.
3. All authentication and authorisation is controlled by central servers.
4. System management is centralised.
5. The objective is that desktops are “plug and play” and require no local support.

The applications run locally to ease any latency problems of running them centrally and the BRA assumes that there is sufficient bandwidth for the data to be held centrally. In addition it assumes that all desktops will be essentially identical, allowing anyone to log on to any machine that they have permission to use. There must be a strong systems management regime in place to keep the desktops' software installations in unison.

The BRA has central configuration and management which will simplify system administration, concentrating all important data on central servers for easy backup and management, and making individual client machines disposable, thus reducing the impact of the failure of such a machine.

Keeping the data locally means that there is an identification of machine with user. This causes problems when the user changes location or leaves the organisation. It also makes the location of the desktop user-specific, which makes concepts such as hot-desking difficult. Holding all data centrally removes these difficulties and makes the use of the desktop much more flexible. It also allows the size of the desktop's local storage to be kept to a minimum.

Making the desktop a plug and play device eases installation and therefore reduces the cost of supporting them.

Many Administrations will already be using a variant on such an architecture for all the reasons given and so the BRA is felt to be a reasonable choice.

The BRA is not appropriate for laptops or for desktops which are not permanently connected to the Administration's network. Such devices would either have to be very fat or have some sort of distributed file system which supports disconnected operation. Such OSS distributed file systems

do exist, such as *CODA*, *OpenAFS* and *InterMezzo*, but they have not been tested by **netproject.**

9. Functional Groups

The reference model is based on functional groups, defining the typical types of non specialised computer activity in an Administration. This means that activities such as Project Management or Geographical Information Systems are not considered. The activities not considered should in general be the ones used by only a small proportion of the user population.

The functional groups are split into Principal and Subsidiary groups. The Principal groups represent functionality that is defined in business process terms. The Subsidiary groups provide support services to the Principal groups and therefore would not normally be implemented on their own.

9.1. The Principal Groups

9.1.1. Office

This is the creation, modification and printing of files containing standard formatted business data such as letters and reports. Also the creation, modification and printing of spreadsheets and presentations. There must be utilities to manage these files. The de facto Microsoft file formats ***.doc**, ***.xls** and ***.ppt** must be able to be read and written with high accuracy. In addition open formats such as PDF need to be both readable and writeable. Local European languages and settings such as currency and alphabet must be supported.

9.1.2. Mail

This is the creation, receiving and displaying of electronic mail including support for secure mail as S/MIME.

9.1.3. Calendaring and Groupware

This is the creation and management of personal and group calendars and address books. The calendars should allow meetings to be arranged and the booking of rooms. The address books should integrate with the other functional groups.

9.1.4. Web Access and Services

This is the ability to access Internet service protocols and display the results. This is normally done with a browser. In addition the ability to create content and make it available as both internally and externally.

9.1.5. Document Management

This is the central storage of documents with efficient retrieval mechanisms.

9.1.6. Database

The manipulation of structured data in both personal and central databases.

9.2. The Subsidiary Groups

These groups are generally defined by technical services and would not normally be implemented on their own. They include:

- operating systems

- File servers
- user management, authentication and authorisation;
- virus and spam detection;
- backup and recovery
- print management.

The complete list is defined in Chapter 12 below.

9.3. General Considerations

Administrations have specific requirements over and above the needs of a standard office environment. Some of these are imposed by local, national or European legislation. In particular:

- They have to be able to accept files from the public in formats which are commonly used. In reality this means at least accepting all of Microsoft's formats, but could also require, for instance, accepting *WordPerfect* and *Lotus Notes* formats.
- Some applications exchange information between members of the public and the Administration, which needs to be done in a secure fashion with notification of delivery.

10. The Reference Model – Summary.

The wide range of OSS available means that for many functions there are a number of different applications available. The choice of which application to use is not always clear cut and sometimes the final choice will be determined by the preferences of the person making the decision.

The reference model used in these guidelines must therefore be treated as an example of a system which is known to work, rather than a recommendation of a system which should be used in all circumstances.

The guidelines discuss the matters which decision makers should take into account and it may well be that they will come to different, but equally valid, conclusions. In any case, local constraints on the Administrator may require a different model to be chosen.

The possible choices for each group is discussed in detail in Chapter 11 for the Principal groups and Chapter 12 for the Subsidiary groups.

There are some useful reference web sites which contain lists of OSS applications, showing what is available and which of them are candidates for replacement of which proprietary applications. http://linuxshop.ru/linuxbegin/win-lin-soft-en/ is an example.

http://www.osafoundation.org/desktop-linux-overview.pdf contains details of many of the applications discussed below and is therefore another place where Administrators can obtain further information.

One of the strengths of OSS is that it is modular and can be put together in many different ways, allowing systems to be tailored to meet specific needs. This modularity is possible because OSS conforms to open and publicly available interfaces. Unfortunately this flexibility can sometimes be its failing, as Administrators can be daunted by the choice available. There are many organisations that can provide help and support, just as there are in the proprietary market.

Note that not all functional groups have reference choices, either because of the lack of relevant case studies or because **netproject** were unable to assess properly the relevant products within the scope of this study.

The detailed Reference Choices for the desktop are listed in Appendix D and for the server in Appendix E. In addition **netproject** have demonstrated a way of installing desktops which makes the installation very simple. The relevant code is set out in Appendix F.

10.1. The Desktop

The operating system is GNU/Linux from the Red Hat 8.0 distribution. The user interface is based on Gnome which is part of the Red Hat distribution, but the Gnome-based Ximian *XD2* desktop is well worth considering. Red Hat, in this distribution, have tried to merge the KDE and Gnome user interfaces.

The filesystems containing binaries (such as **/usr**) are mounted “read-only” to stop users altering their contents and the remaining ones are mounted “noexec” to stop code being run from them. To enforce this, the user interface should not allow users to run programs other than through predefined interfaces. This means that command line access or the ability to create or change menu items or icons has to be removed. The machines should have no floppy or CD drives thereby restricting (but not stopping) other filesystems being attached locally.

The filesystems containing volatile user-based data are mounted from a central NFS server. User authentication is done against a central LDAP database.

Central DNS servers do IP address and name resolution and a DHCP server provides the

desktop's network configuration details at boot time.

The major desktop functions are delivered as follows:

10.1.1. Office

OpenOffice.org is chosen because:

- If the Administration is migrating from a *Windows* environment, *OpenOffice.org* can be run on *Windows*, giving users initial contact with the new software in a familiar environment.
- It has some of the best Microsoft file format integration.
- It is becoming the de facto alternative to *Office*.

10.1.2. Mail

Evolution is chosen because it has a very similar user interface to *Outlook* and so will be easy to learn for many people. It also has some very useful features such as Virtual Folders. However *Evolution* does not connect to *Exchange* version 5.5 (although apparently one is planned) and, so if the Administration is using this version, a web based OSS groupware solution will be necessary at the moment (unless a proprietary solution is chosen). *Kmail* on the other hand supports S/MIME whereas *Evolution* does not (although there are plans to do so) and *Kmail* has recently been developed to run as a client against the *Kgroupware* groupware server. The choice is finely balanced therefore and depends upon the immediate needs and the current setup.

10.1.3. Calendaring and Groupware

Evolution is chosen for personal calendaring and contact management. Groupware is difficult with OSS at the moment, only web based solutions are really available although recently the *Kgroupware* project has produced a solution using *Kmail* as the client. Hence for an truly OSS solution then a browser would be used for groupware access.

10.1.4. Web Access

Galeon is chosen because it is a fast, single-function browser that has a nice user interface. *Mozilla* is an alternative if a full product including mail reader and address book facility is required. *Mozilla* would also be the choice if the Administration is currently using *Windows* desktops, and the new browser is required to run in the existing environment to allow users initial contact with the new software in a familiar environment.

10.1.5. Document Management

A web based content management system such as *Aswad* would be the choice. However, the *Aswad* project seems to have stopped and so another choice is now needed.

10.1.6. Databases

Personal databases are either based on *MySQL* or a web-based groupware product such as *phpGroupWare*.

10.2. The Servers

The operating system is GNU/Linux from the Red Hat 8.0 distribution. This choice would be different for highly secure machines such as firewalls, where OpenBSD would be used in conjunction with GNU/Linux.

The major server functions are delivered by:

10.2.1. Mail

The ***MTA*** (Mail Transport Agent) is *Exim*, because it is a fully-fledged product comparable to *Sendmail* in scope, but easier to maintain. It also understands *Sendmail* options can therefore be run as a compatible replacement for *Sendmail*. *Postfix* would be an acceptable alternative.

The ***MAA*** (Mail Access Agent) is *Courier-IMAP* which was felt to be easier than *Cyrus* because it has a simpler mailstore. However *Cyrus* would be a good choice.

10.2.2. Calendaring and Groupware

PhpGroupWare or the *Horde* would make very good web based solutions. The new *Kgroupware* has not been evaluated.

10.2.3. Web Services

Apache is chosen because it is the market leader with a wide range of associated tools and support. Other servers could be used for specific tasks; for instance, *Zope* (see 11.4.2 below) could be used for content management.

10.2.4. Document Management

Now that *Aswad* (see Section 11.4.3) seems to have stopped there is no reference solution. The discussion in Section 11.5 indicates that there is some choice available here.

10.2.5. Databases

For large databases which are principally read-only, *MySQL*; for other kinds of databases *PostgreSQL*.

11. Applications – Principal Groups

11.1. Office

The current de facto standard is *Office* which includes *Word*, *Excel*, *PowerPoint* and *Outlook* together with their associated file formats: ***.doc**, ***.xls** and ***.ppt**. These formats are not open, and change from one version of *Office* to another. Even Microsoft's own products cannot guarantee the ability to read and write a file with 100% accuracy unless the files was created with the same version of their product.

OSS applications are now able to read these formats with enough accuracy that any problems encountered are not dissimilar from those found in using different versions of Microsoft's own products. The older the format, the better the OSS applications are at dealing with it. OSS applications tend to be better at reading files in Microsoft's formats than writing files in those formats. In general the OSS applications can therefore be used with confidence. For example, Ximian, with their latest desktop product, have set the default file formats in their version of *OpenOffice.org* to be Microsoft's.

The exception is where some sort of collaborative working is required and at least one of the parties insists on using a proprietary format. The reading, alteration and re-writing of files in these formats can introduce anomalies which the use of a single proprietary application would not. However it must be borne in mind that this sort of corruption can also occur if different versions of the proprietary software is used.

For files that are only intended to be read and not updated, then PDF format should be used as discussed in Chapter 7.3 above.

Some may feel that the user interface should be as similar as possible to the Microsoft's software to minimise re-training costs.

Templates and *Visual Basic* (VB) macros are common in many Administrations. These are similarly in a closed proprietary format and will need to be re-written.

There are three OSS office suites, *OpenOffice.org*, *KOffice* and *Gnome Office*, all of which should be considered.

A pilot study comparing the various OSS office suites' ability to handle Microsoft *Office* files is available on the web:

http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=55

11.1.1. *OpenOffice.org* and *StarOffice*

OpenOffice.org is an OSS office suite based on *StarOffice*, which was produced by a German company called StarDivision. Sun Microsystems purchased StarDivision and contributed the code to the OSS community. It continues to market a version of *OpenOffice.org*, still called *StarOffice*, which it sells at a much lower price than comparable proprietary packages.

StarOffice and *OpenOffice.org* are essentially identical except that:

- Sun Microsystems provides commercial support for *StarOffice*.
- *StarOffice* has an inbuilt database system (*Adabas*).
- *StarOffice* has some extra filters for migration to/from other office packages (however the *Wordperfect* filter is not available on GNU/Linux for licensing reasons).

- *StarOffice* has some proprietary fonts.
- *StarOffice* is only available in a rather more restricted set of languages (Dutch, French, Italian, English, German, Spanish and Swedish).
- *OpenOffice.org* is updated more frequently than *StarOffice*.

Both applications are comparable with *Office* but do not include an email client. Both handle most *Office* files up to and including *Office XP*, although compatibility gets progressively worse with *Office* versions later than *Office 97*. They do not handle files that are password protected (except sheet level protection in spreadsheets) and have some problems with OLE linked graphic objects.

They have their own implementation of Basic and cannot handle *Visual Basic* macros, which must be translated by hand. Although the translation is a migration cost, the lack of macro support stops macro viruses being transmitted. Sun Microsystems are setting up links with companies to translate Microsoft's macros and templates to a form compatible with *StarOffice*.

They also offer a *Java* interface, but currently only recognise the Sun Microsystems JDK. Sun Microsystems have announced a project to develop a *Visual Basic for Applications* (*VBA*) to *Java* translator.

The downloadable version of *OpenOffice.org* only contains a few dictionaries but dictionaries are available for most European languages. Pre-built versions exist already for 25 different languages.

Although *OpenOffice.org* does not currently offer a database package, it does have ODBC and JDBC interfaces to many common database systems including the popular OSS ones. Also there are no conversion filters for *Wordperfect*, but this is planned for a future release.

Both run on a range of operating systems including *Windows* and GNU/Linux.

Reviewers are starting to rate *OpenOffice.org* for both functionality and stability – see:

http://www.raycomm.com/techwhirl/magazine/technical/openofficewriter.html

The following site provides a good *OpenOffice.org* manual:

http://www.taming-openoffice-org.com/

11.1.2. Koffice

This is the office component of the KDE desktop. It is an integrated package providing word processing, spreadsheet, charting, presentation, illustration, report generation and flow charting tools with an optional desktop called Workspace.

The Microsoft file filters are not as good as the ones provided by *OpenOffice.org*. It does not have a macro language, but scripting is available.

Koffice works well and has a nice intuitive interface.

11.1.3. Gnome Office

This is a collection of programs that are written to the Gnome standards and thus can integrate with one another, have a similar user interface and should be able to embed one another.

OpenOffice.org is now considered part of *Gnome Office*, even though it does not

adhere to the Gnome standards. Ximian, in particular, are working to make *OpenOffice.org* more compatible with Gnome and have included their own version in their latest desktop product, *XD2*. See http://www.gnome.org/gnome-office/ for more details.

Gnome Office has many components including *AbiWord* (wordprocessing), *Gnumeric* (spreadsheet), *Sodipodi* and *Sketch* (vector graphics drawing), *Gimp* (image editing), *Eye of Gnome* (image display), *Dia* (vector graphics, similar to *Visio*) and *Agnubis* (presentation graphics).

These components are at varying degrees of usability; for instance *Abiword* is very good at basic wordprocessing but has problems with tables, and *Agnubis* is very limited, whereas *Gnumeric* is a very capable spreadsheet. Development work on nearly all the components is still very active.

Gnumeric in particular have the goal of developing a spreadsheet which can do everything *Excel* can, and a great deal else as well. The latest beta test release of *Gnumeric* (1.1.20 at the time of writing) supports all of the worksheet functions of the US version of *Excel*, and a production release including this level of functionality is expected in September 2003. The developers come from a financial background and they have included a number of features which make *Gnumeric* particularly useful for financial applications. It is in this area that they believe *Gnumeric* excels over *Excel*.

Gnumeric uses ***.xls** as its native file format, while *OpenOffice.org* converts spreadsheets to an XML-based format.

The range of products available is interesting and, together with *OpenOffice.org*, provide a number of different solutions. However if an integrated suite is required then currently *OpenOffice.org* is the only real solution.

11.2. Mail

Mail is a complex area with several logical components, and has a wealth of OSS applications some of which provide overlapping functionality. It is also closely linked with other issues including virus and spam ("junk email") control.

The choice of appropriate applications is complex, and a detailed discussion of the issues is included in Appendix C together with a definition of all the terms used here.13.3

11.2.1. MTA

The major OSS MTAs are *Sendmail*, *Exim*, *Courier-MTA* and *Postfix*. There are many others but these are regarded as the main ones for large scale use.

Traditionally Unix and OSS sites have used *Sendmail* as their MTA. Unfortunately it has had a poor security record and is also notoriously difficult to configure.

All the others have good reputations and at a technical level there is little to chose between them. However, there is a significant difference in the standard of documentation available, with *Postfix* being better documented at the beginner level and *Exim* at the expert level.

Exim and *Postfix* are included in some OSS distributions but may not be the default.

Courier-MTA comes as part of a family with a MTA, ***MDA*** (Mail Delivery Agent), MAA and webmail package (*sqwebmail*) available. Each part can be used on its own, or integrated with the rest of the family.

Qmail is an MTA which is often mistaken for OSS. Although source code is available, it has a

copyright licence which allows only the distribution of versions exactly identical to those distributed by its author; it is therefore proprietary software. Its restrictive license makes it extremely difficult for OSS distribution vendors to support it or to integrate it with their other software, and they therefore do not distribute it. However, it has a good security record and is widely used.

Another interesting product is the Apache *James* mail server. This is intended to be a fully fledged MTA written in *Java*. It lacks some features at the moment but is worth keeping an eye on for the future.

The reference choice is *Exim*. This because it is as capable as *Sendmail*, while being easier to configure and probably more secure. The other two are perhaps less able to deal with large volumes of messages. The choice is not clear cut and Administrators should really make their own decision based on local requirements.

11.2.2. Mailstore

Most Administrations would want clients to use centralised mail storage rather than downloading messages to local storage on the desktop client. For this reason we strongly recommend the use of IMAP.

There are three well-known OSS IMAP servers; *UW-IMAP* (sometimes just called IMAP), *Courier-IMAP* and *Cyrus*.

UW-IMAP has a poor security history and we do not recommend it.

Of the other two, *Courier-IMAP* is widely considered to be the easier to configure. It has a smaller footprint and works well with *Postfix* and *Courier-MTA*. It is the MAA part of the Courier family. It requires maildir as the mail storage format. It is reputed to have difficulty handling some S/MIME messages, moving headers around in a way that invalidates the signature.

Cyrus uses its own mail storage format that is similar to maildir and requires its own MDA to populate the mailstore.

Both *Courier-IMAP* and *Cyrus* support ***TLS*** (a standard authentication and privacy protocol).

There are a number of MDAs, for instance *procmail*, Courier's *maildrop* which comes as part of the Courier family and *Cyrus*' *deliver*. The MDAs also have the ability to filter mail according to sophisticated rules which is useful if the MUA used does not have filtering facilities.

The reference choice was *Courier-IMAP* with no MDA. An MDA is not needed, because *Exim* is capable of writing directly to maildir structures and *Evolution* has its own very capable filters.

11.2.3. MUA

There are a large number of text-based and GUI MUAs available in the OSS field.

For those who are used to *Outlook* or *Outlook Express* and want to have something similar, the obvious choice is *Evolution*. *Evolution* is not only a mail client but also a Personal Information Manager (PIM). It has LDAP integration and can therefore access Administration name and address data as long as it is held in *Evolution*'s own schema. It is being very actively developed by Ximian as their flagship product. Ximian make a product called *Connector* which allows *Evolution* to connect to *Exchange* (but not version 5.5).

Connector is proprietary and has a licence fee.

Evolution unfortunately doesn't fully support disconnected mode IMAP. It only copies certain mail to the client rather than all. However it does have a very good feature called "virtual folders" which allows the user to define rules for viewing their mail in many different ways without actually having multiple copies of the mail.

Alternative MUAs are *kmail* and *sylpheed*. Both are very good and integrate with major OSS desktop environments. *Kmail* would be used if KDE is the desktop, while *sylpheed* would be used if Gnome is the desktop.

Evolution supports *GPG* but not S/MIME although it is expected to do so soon.

Mozilla supports S/MIME but not *GPG* or PGP although it will do soon.

Kmail supports S/MIME, *GPG* and *PGP* as part of the Ägypten project funded by the German Government.

Many groupware packages also include IMAP and POP3 compatible clients. In general they are not as good as *Evolution*, but may be sufficient if they integrate well with the other groupware functions.

In some cases it may be best to migrate certain categories of mail user to a web-based user interface. *SquirrelMail* is a particularly good one, which can be found at http://www.squirrelmail.org/.

The Open Systems Applications Foundation produce a product called *Chandler* which is in its infancy but is worth monitoring for the future. It is a potential competitor for *Evolution*.

Since the majority of users will probably require a Microsoft *Outlook* look-alike, *Evolution* is likely to be the preferred choice, and it is the reference choice. However if S/MIME is required immediately then *Kmail* must be used which may mean using *KDE* instead of *Gnome*.

11.2.4. Anti-Virus

If OSS systems are correctly configured then viruses have limited effect. However there is the problem of passing viruses on to sites that run other operating systems, including Microsoft's products in particular.

There is one OSS anti-virus product, *ClamAV*, but it has some problems. Therefore fully supported proprietary products are recommended as present.

The best way to run such products is as part of the MTA. Both *Postfix* and *Exim* provide means of incorporating such filters.

There is a choice of proprietary products. *Sophos*, *RAV* and *Vexira* all have good reputations. *Trend* is good at virus checking but must be run with super user access, which makes it a greater potential security problem should it be compromised.

No choice is made for the reference model as **netproject** have been unable to fully test the products.

11.2.5. Other Tools

There are a number of anti-spam tools and tools that prevent executable attachments being downloaded with email.

SpamAssassin is probably the best known of the message analysis anti-spam tools.

Anomy Sanitizer is a configurable tool capable of removing attachments from messages. It is configurable, but needs to be used with caution since the removal may invalidate the signature of a signed document.

MailScanner is a general framework for content checking, including anti-virus and anti-spam measures. It can invoke one or more proprietary anti-virus products on email messages and can also use *SpamAssassin*, as well as applying its own rules.

Fetchmail collects email from a remote mailstore and either stores it or passes it to another MTA. Because it pulls mail onto a machine (i.e., the transfer is initiated by the receiving party) it is useful where, for security reasons, Administrators do not want to open a port on their Internet gateway machine to allow email to be pushed to them (where the sender initiates the transfer) as would happen with the normal SMTP model.

OfflineImap is a tool which allows a mailstore on a central server to be synchronised with a mailstore on a client. This is done by the client connecting to the server using IMAP on a regular basis. The local structure is maildir. This can be very useful to allow disconnected mode IMAP to be emulated if the MUA doesn't fully support it.

Whoson allows the POP-before-SMTP method of authentication of remote users. The use of this is necessary if users are to be able to send email through the Administration's servers remotely if authenticated SMTP is unavailable and the Administration's MTA is open to connections from IP addresses outside its trusted range.

11.2.6. Problems Experienced

Storing data in an LDAP server requires a schema to be chosen. The schema must be compatible with all the clients which may require to access the data. Fortunately some packages come with a schema that not only supports their needs but also the needs of several other packages as well.

Courier-IMAP comes with a schema but *Exim* does not. The *Courier* schema was found to support *Exim* as well. We do not know if it supports all the capabilities of *Exim*.

Some problems were discovered with the LDAP configuration file for *Courier*. Fixes have been fed back to the developer but have not yet been included in a release.

Using *Courier* with *Whoson* requires some patches to *Courier*. Some were available on the *Whoson* site, but were rather old and needed significant updates to work with the selected version of *Courier*.

11.3. Calendaring and Groupware

Calendaring is an ill-defined subject within OSS. This is due to the absence of open standards for communication between the clients and the central server. Hence the products developed so far use web-based delivery and this may not provide the look and feel that people have become used to with *Exchange* and *Outlook*. This area is a significant weakness in the OSS portfolio.

Products listed in the table overleaf can be assumed to use web-based delivery unless stated otherwise. All of them are part of groupware suites which have a wide variety of other features.

Most of the groupware products are written in PHP or Perl, and can therefore be customised. Some interesting integration of facilities has been built into these products.

phpGroupWare (http://www.phpgroupware.org/) has a good reputation.

Groupware Product Details

Product	*Email*	*Calendaring*	*Document Mgmt*	*Chat*	*Task List*	*Contact Mgmt*	*Database*	*Time Sheet*	*Scheduling*	*Other facilities*	*Remarks*
NullLogic	Y	Y	Y	[1]	Y		Y				
Twiki		Y	Y	Y			Y				More of a framework than a product.
phpGroupWare	Y	Y	Y		Y	Y	Y		Y		Difficult to find precise information from the web site
phProject	Y	A, B	Y		[2]	Y		Y		Project Management, Bookmarks, Reminder, Search System, voting system, request tracker	
Tutos	Y	B	Y			Y		Y			
Twiggi	Y	Y			[2]	Y			Y	To Do, Bookmarks	

NOTES

1 Standard IM plus forum (BBS)

2 Short “sticky note” type functionality

A Timesheet recording; Request Tracking, a form of workflow management; “ToDo” list; Dynamic Projects

B Resources (e.g. meeting rooms, overhead projectors) and Event recording (forthcoming as well as past)

The *horde* is a framework for running other applications. For instance, *Imp*, a web mail server, *Turba*, a contact manager, and *Kronolith*, a calendar. See http://www.horde.org/.

NullLogic appears only to offer English language interfaces but *phProject*, *Tutos*, *Twiggi* and *Twiki* all support a range of languages.

A very recent OSS product is *OpenGroupware* from http://www.opengroupware.org/. This is the formerly proprietary application *SKYRiX* which has been made OSS by its owners. This is intended to be an *Exchange* substitute. There has been no time to fully investigate it yet but initial indications are that it will become very influential.

Another recent product is *Kgroupware* (See http://kolab.kroupware.org/). This product has a client based on *Kmail* is worth investigation particularly if *KDE* is chosen as the user interface or *Kmail* is chosen as the MUA to support S/MIME.

11.3.1. Personal calendars and agendas

All the products have the ability to maintain personal calendars and to-do lists unless stated otherwise.

11.3.2. Group calendars

Tutos, *Twiggi* and *NullLogic* all support group calendars. *Tutos* is controllable in levels ranging from individual, through work group and project group to everyone.

In *NullLogic*, calendars cannot be kept private from other members of the group, but tasks can.

11.3.3. Meeting organisation

Many of the products incorporate resource scheduling features which can be used to plan meetings.

Tutos allows for automatic allocation of people, together with automatic email notification to those who are not on the shared calendar (such as those in other organisations). It keeps track of acceptances and will send reminders via email if required. *phProject* is similar, and will handle ***SMS*** text notifications as well.

NullLogic supports all the above features, except for room allocation.

11.3.4. *PDA* synchronisation

phProject has an add-on that synchronises with PalmOS-based PDAs. PDA synchronisation is also supported as part of Gnome and *Evolution*. Most popular PDAs can be synchronised.

11.4. Web services

11.4.1. Browser

The main OSS browsers are *Mozilla*, *Galeon* and *Konqueror*. There are others such as *Lynx* which is text only, and often used as the basis for browsers for people with disabilities, and Mozilla *Firebird* (formerly known as *Phoenix*), a light weight variant of *Mozilla*. There is also a proprietary browser, *Opera*, which has a version which works on GNU/Linux. *Netscape* is based on *Mozilla* and runs on OSS platforms, but includes some proprietary code.

Mozilla is the major OSS project based on code released by Netscape, and is the basis for *Netscape 7*. It contains mail and news components together with an address book and a web authoring tool. Much of the *Mozilla* code is used by other projects including *Galeon* and *OpenOffice.org*.

Galeon is a browser only and is designed to be small and fast. It is based on *Gecko*, the rendering engine upon which the *Mozilla* project is based, together with a Gnome user interface.

Both *Galeon* and *Mozilla* support all web-related open Internet standards and can execute properly-written Java and Javascript.

Some content requires a plugin that is only available for *Windows*, such as *Shockwave Director*. The proprietary CodeWeavers *CrossOver Plugin* product allows plugins that work under *Windows* to run under GNU/Linux.

Konqueror is the web browser written for the KDE desktop and is also used as a drag-and-drop file manager. It is based on the KHTML rendering engine, with the Mozilla *Gecko* one as an option, together with a KDE user interface.

11.4.2. Web servers

The most popular OSS web server is *Apache*, which, according to the Netcraft survey (http://www.netcraft.com/) has over 60% of the market and its share is growing. An increasingly popular combination of products goes under the name LAMP: *Linux*, *Apache*, *MySQL* and PHP. This provides a framework for web sites accessing SQL databases via the PHP language. All the components are OSS.

The Apache project contains a number of sub-projects, one of which is called *Jakarta* and covers the server-side use of Java. *Jakarta* itself consists of sub-projects, two of which are *Tomcat* and *Slide*. *Tomcat* provides a product for Java servlets conforming to the JSP standard, and the ability to use technologies such as IBM's *Websphere*. *Slide* is a Java-based implementation of WebDAV allowing content management. See http://www.apache.org/ for full details.

Other OSS servers worth considering are *Zope* and *Tux*.

Zope (http://www.zope.org/) is designed to provide dynamic web content support and is based on an object-oriented model. It is an interesting package, as it combines a content management system with a web server and a template system in one package. *Zope* also supports modular add-ons (called products) and is based on the object-oriented Python language. It is common to find *Zope* deployed “behind” *Apache* in a multi-server configuration, where *Apache* serves static content and acts as a cache-based accelerator for the parts of the site managed by *Zope*.

Tux is a Red Hat development now called *The Red Hat Content Accelerator*. It uses a special kernel and is supposed to provide very fast response for static pages.

Roxen is another combination web server and content management system, although for full management functionality it is necessary to buy proprietary add-ons.

Of these, *Apache* is by far the most popular. It currently runs 63% of the world's public websites and is gaining market share from *IIS* steadily, so there is plenty of experience to draw on when planning a migration. *Apache* is a modular server, with a core protocol engine and a large selection of modules for specific purposes.

11.4.3. Portal / Content

Zope, together with other OSS components, are part of a EU funded project called *ASWAD* http://www.aswad-project.org/ which is designed to provide content management. **netproject** had hoped to be able to assess this project but have had difficulty obtaining up-to-date details. An interesting project based on *Zope* is *Plone* (http://www.plone.org/).

JBoss (http://www.jboss.org/) is a Java-based application server is. It has a good reputation and is actively developed.

There are now many OSS content management products, as a visit to http://www.oscom.org/matrix/index.html will show. **netproject** could not find any detailed information in the published case studies and were unable to investigate these in the detail needed to choose a Reference Model candidate.

11.5. Document management

11.5.1. Registration and retrieval

Document Management can, and perhaps should, be thought of as a form of content and workflow management. This is the sort of thing that *ASWAD* (see 11.4.3 above) is intended to cover. **netproject** would therefore recommend that a solution based on the available content management solutions is adopted. In particular, those using WebDAV might provide the most useful solutions.

A German standard called DOMEA (Disposition and Archiving of Electronic Records) which isn't widely used outside of Germany has been adopted by IBM in conjunction with *SAP*. Most of the documents relating to DOMEA (in particular, those found by a Google search) are written in German. There is a company called FabSoft which provides support for DOMEA on GNU/Linux on IBM mainframes. There appears to be no OSS product which supports DOMEA.

Some of the groupware products provide support for document management:

- *Tutos* includes a document management system that has version management as well.
- *NullLogic* includes a simple capability to store, index and download files. It does not seem to offer a change management system. It has a generalised query mechanism that could be set up to offer sensible indexing.

11.5.2. Collaborative working

This function can be implemented ad-hoc by simply exchanging documents between people. Exchange can be by email attachment or by mechanisms such as those used by CIRCA.

Collaboration requires parties to agree on the format of the document, and currently many people use Microsoft's ***.doc** by default. This default means the parties have to trust each other, because such formats can be efficient carriers of viruses. Also, as a standard, ***.doc** is not ideal because it is constantly changing; the format as used by *Office 2000* is not identical to that used by *Office 97*. This means that the parties also have to agree on which software version to use.

There is pressure to adopt standards-based document formats, in particular ones based on XML. *OpenOffice.org* provides an open XML-based document standard which could be used as a basis for collaboration.

A more structured approach would be to adopt a content management/workflow solution as described above.

The *Tutos* groupware product allows documents to be subject to the control of just one person or by all within a defined group. *NullLogic* and *Twiki* also have refined controls.

As for content management above, there is no Reference Model candidate.

11.6. Databases

11.6.1. Central databases – application-based

The OSS database systems available include *MySQL*, *PostgreSQL* and *Firebird*. They have significantly different characteristics and applicability.

MySQL is a lightweight SQL database favoured for web servers and similar applications. It is appropriate in situations where reading predominates over writing. It does not support database procedures or complex sub-queries. Support for database procedures is planned for version 5.

PostgreSQL is a full-featured DBMS comparable with *Oracle* and *DB2*, but without the more advanced features needed to handle very large volumes of data.

Firebird is a version of Borland's *Interbase* database released under an OSS licence. Much of the code is common with *Interbase* and as such it must be regarded as mature. There is a project to add a database capability to *OpenOffice.org* using *Firebird*, but this is in its early stages.

11.6.2. Personal databases held centrally or locally

Ad hoc personal databases are not well supported in OSS. There is no direct equivalent to *Access*, nor is one being developed.

Several of the groupware packages do offer some capability in this area using a variety of OSS SQL databases as a back-end. In some cases (such as *NullLogic*) ordinary users can only use pre-defined queries. Some offer the ability to define forms that can be used to store and access data.

11.6.3. Database Connectivity

Most DBMS products support direct APIs with C language bindings. Some also support C++ natively.

All offer ODBC and JDBC connectivity. Some also offer .NET connectivity.

There is a product called *Unix-ODBC* which provides ODBC-like connectivity to Unix and GNU/Linux programs.

11.6.4. Performance

Database performance is heavily dependent on the size of tables involved and the complexity of queries.

None of the OSS offerings would be able to cope with the demands that proprietary databases such as *Oracle* can. This is especially so in applications such as data warehousing, partly because none of them yet have the ability to provide a distributed database capability.

The proprietary products *Oracle*, *DB2*, *Ingres*, *Informix*, *Progress*, *Mimer* and *Sybase* are available to run on GNU/Linux and could be considered the preferred options for heavy database applications where the OSS products are not yet suitable. The *Oracle* development tools are supported on GNU/Linux.

12. Applications - Subsidiary Groups

12.1. Operating System

There are several OSS operating systems, including *OpenBSD*, *FreeBSD*, *NetBSD* and various "Distributions" (explained below) of GNU/Linux, although, of these, many people have only heard of GNU/Linux, and have generally heard of it by the name Linux.

An operating system consists of a kernel which runs in supervisor mode, along with supporting programs running under the control of the kernel in user mode. *Linux* is a kernel but it requires supporting loaders, compilers, drivers etc. Most of these supporting programs are provided by the Free Software Foundation's GNU project and so the totality ought to be called GNU/Linux, which is the term used in this report.

The *Linux* kernel is provided packaged together with a set of supporting programs and applications by a number of companies such as Red Hat, SuSE and Mandrake as a Distribution. The contents of a Distribution should interwork, and the kernel may well be patched with changes not available with other Distributions. The choice of Distribution must therefore be considered, as each has its strengths and weaknesses.

There are other Distributions such as *Debian* and *Gentoo* which are not prepared by a commercial organisation and this has implications for the way in which support is provided. Support for these distributions comes either from third parties or from access to mailing lists on the Internet. Both of these can provide acceptable levels of cover.

Debian has a reputation for solidity and its stable section contains code which has been thoroughly tested by many people world wide. There are also two other sections providing increasing levels of leading edge software. The stable branch also has the reputation of being out of date. This is unfair to some extent because most commercial users are principally interested in stability and lack of bugs, and not whether the latest peripheral can be supported.

Gentoo is a source-only distribution, which means that the Administration can build its own binaries easily, tailoring the Distribution to their environment and hardware. Building such a distribution from scratch is time consuming but once the binaries have been built they are available generally. This is a new distribution and is worth considering. Because most other Distributions are supplied with full source code, it is possible to tailor any of them the same way; *Gentoo*, however, may be more amenable to such treatment.

The commercial Distributions come in different packages with different levels of support available. The Distribution available via the Internet is invariably only supported for about a year; users are then expected to upgrade. Most companies provide an "Enterprise" version which is guaranteed to be supported for 5 years or more and which is based on stable versions. Such versions also have a support contract associated with them which is sometimes called a licence, even though the code is licensed using GPL or LGPL and may not be licensed otherwise. It is the availability of such supported and stable distributions that many Administrators want. Indeed one reason to move to OSS is the lack of pressure to upgrade constantly and unnecessarily. The companies promise to backport bug fixes. Red Hat, for instance, have their Enterprise Range which consists of three products, two for servers and one for technical workstations, each of which appears to be based on version 7.3 of *Red Hat Linux*, even though the current downloadable version is labelled 9.

Our opinion is that GNU/Linux is the preferred platform for workstations as it offers better configuration tools and a variety of packaging setups that are more suitable for desktop use. Also, some popular desktop products do not work on all of the alternatives (for example the

Mozilla web browser currently does not work on *OpenBSD*).

For servers the situation is much less clear cut. *OpenBSD* has by far the best security record of any OSS operating system, being able to claim only one remotely exploitable vulnerability in 6 years. It should be the preferred platform for anything that requires higher than average security (such as firewalls and De-Militarised Zone servers).

Server applications generally run well on all of the BSD platforms and GNU/Linux, although many have been written for GNU/Linux and are then ported. Proprietary software is often only available for GNU/Linux.

12.2. User Interface

12.2.1. Desktop manager – look and feel

There are several choices, ranging from very simple, light-weight window managers like *icewm* to fully featured session managers like those included in *Gnome* and *KDE*. The choice should depend on the intended use.

Of the session managers *KDE*'s is the more mature but *Gnome* is catching up fast. *Gnome* is being supported by Sun Microsystems and members of the Gnome Foundation. **netproject** considers that it has a better architecture and believes it has a better future.

A *Gnome*-based desktop has been issued recently by Ximian called *XD2*. This works on top of a number of different base Distributions including *Red Hat* and *SuSE*. Ximian have paid particular attention to integrating the various different applications to make sure that they work in a similar way. This means that they have included their own versions of some products like *OpenOffice.org*. It is too early to comment fully on this desktop but the initial impressions are promising.

The choice for any Administration is likely to be one of personal preference; both environments are very capable. Applications designed to work in one environment will work in the other but more specific features such as session management may not work properly.

12.2.2. Language

The desktop managers offer most European languages but may be patchy in their support for any given language.

12.3. Security

All functional groups must be configured with security in mind. Security at the software level can only work if it exists in the wider framework of security management. **netproject** has not fully investigated all the functions given as part of this report.

12.3.1. Data encryption

12.3.1.1. Data in transit

Confidential data on internal LANs should be encrypted where ever possible. Sensitive data sent over the Internet should always be encrypted. This can be done by tunnelling connections over products like *ssh* and *stunnel*.

12.3.1.2. Data in storage

Confidential data held on mobile devices should be encrypted on disk. The ideal is that all data should be encrypted but this would impose significant overheads which will not always be acceptable. There are a number secure filing systems and the next Linux kernel provides better support for them. For instance http://koeln.ccc.de/archiv/drt/crypto/linux-disk.html has a discussion of the various methods available.

12.3.2. Authentication

Secure methods to identify uniquely a person or machine which is part of a communication with other persons or machines. This includes signatures and PKI infrastructures. No PKI systems were tested as part of this project. All authentication was done against an LDAP database using standard password challenge.

12.3.3. Authorisation

Once authenticated, to determine what a person or machine can do and in what circumstances. This is normally part of the operating system or application code. Role Based Access Control or RBAC has been defined by the NIST in the USA and is available for Linux. (See http://csrc.nist.gov/rbac/).

12.3.4. Virus control

Virus control is needed principally to stop onward transmission of viruses to other non-OSS sites. Although email is one of the main ways that viruses are transmitted, it is not the only one, so generalised file scanning is needed to stop transmission by other means.

Unfortunately we know of no OSS product which does such scanning. However by properly configuring the filesystems on both servers and desktops it is possible to ensure that the only executable files are those which were installed by the system administrator. It is therefore important that system administrators make sure the files they install are trustworthy, for example by checking the Distribution vendor's signature on the files.

12.3.5. Proxy Server

A range of intelligent or semi-intelligent OSS proxy servers is available.

Of the web proxy servers, *squid* is the most popular. It has an associated product (*squidguard*) that prevents access to a list of banned web sites.

12.3.6. Firewalls

All current OSS operating systems have internal packet-filtering firewalls, the majority of which are stateful. Stateful firewalls are those which maintain information about ongoing connections and data streams through the firewall, and allow packets to pass which are associated with those connections while filtering out packets which are not. Firewalls which are not stateful examine each packet on its own merits, without keeping any record of previous packets. Specialised plug-ins are available for protocols such as ftp and telephony H.323 which use non-standard means of connecting.

iptables, currently included with GNU/Linux, and *ipfilter*, included with *FreeBSD* are both good firewall products. *Packetfilter* is now included with *OpenBSD* also has a good reputation. Good practice for external firewalls is to have two different ones between the public network connection and internal servers. We do not recommend a single example.

12.3.7. Virtual Private Networks (VPN)

12.3.7.1. OpenVPN

Available for most Unix flavours, this is a mature and powerful offering. Features include public key encryption, dynamic compression for bandwidth management, and the ability to work with NAT (Network Address Translation). See also http://openvpn.sourceforge.net/ for more information.

12.3.7.2. FreeSWAN

This is a GNU/Linux implementation of the IPSEC and IKE standards, meaning it will inter-operate with compliant devices, including special routers and other operating systems. Since IPv6 supports IPSEC natively, *FreeSWAN* might be preferred if this newer standard is used. To benefit from *FreeSWAN*'s unique "Opportunistic Encryption" extension, which can automate security, DNS records must be updated which could be limiting. **netproject** understands that IPSEC may also have problems with NAT. See also http://www.freeswan.org/ for more information.

12.3.7.3. CIPE

This is less mature than the other two and public-key support is still experimental. However it can work with NAT, is available for *Windows*, and ships with *Red Hat Linux* (you can even configure it with their intuitive Network Device Control tool). More information is available at http://sites.inka.de/~W1011/devel/cipe.html.

12.4. Management

The site http://www.infrastructures.org/ provides considerable detail on how to manage a network of machines, both servers and desktops, and has a number of OSS tools for a range of system maintenance tasks. It is maintained by someone who has done these tasks for many years. **netproject** agrees with almost all of the content except, primarily, the section on printer management.

The site shows that Unix, and by extension GNU/Linux, management tends to be done by tools that are put together from smaller single-function units. This modular approach is extremely powerful and it is what allows Unix and GNU/Linux system administrators to be very efficient and effective. It also means that the market for toolkits is small, since system administrators each tend to build their own set of tools.

There are proprietary tools available such as *Tivoli* from IBM and *Unicenter* from Computer Associates.

12.4.1. User management

The maintenance of users and groups of users, including password management. Products like *Directory Administrator* and *gq* allow LDAP databases to be maintained.

12.4.2. Configuration management

Although a well designed centrally-managed client should only have minimal local state, updating its configuration without re-installing from scratch is still desirable for large networks expected to be active for some time. For example, if a core central service is changed, clients may have to be re-configured to use it.

12.4.2.1. Manual Configuration Maintenance

Administrators can maintain configuration updates manually as they could software updates. However, similar synchronisation problems apply. Manual modification of configuration files, often stored in plain text files, is particularly prone to typing mistakes.

12.4.2.2. Cfengine

The *GNU Configuration Engine* (http://www.cfengine.org/) automates remote configuration of networked clients. It supports a wide variety of UNIX flavours, and its powerful class concept allows different groups of clients to be managed with minimal set up. Autonomous agents on the clients can maintain text files, network interfaces, file links and permissions, temporary storage and mounted file systems.

Some of the primitives which can be automated using *cfengine* are:

- Check and configure the network interface.
- Edit text files.
- Make and maintain symbolic links, including multiple links from a single command.
- Check and set the permissions and ownership of files.
- Tidy (delete) junk files which clutter the system.
- Systematic, automated mounting of filesystems (on Unix).
- Checking for the presence of important files and filesystems.
- Controlled execution of user scripts and shell commands.

Cfengine follows a class-based decision structure.

12.4.2.3. System Configurator

System Configurator (http://sisuite.org/systemconfig/) is part of the *System Installation Suite*, and is used by *System Installer*. It can set up and maintain many components of a GNU/Linux installation across many distributions such as networking, storage, time zone and booting.

12.4.3. Software management

This section covers system maintenance of clients from initial setup on new hardware to ongoing updates of software and configuration, and some technologies available to ease their management.

12.4.3.1. System Installation

System installation is the initial setup of software and configuration necessary to maintain a machine. Factory built machines might have no operating system at all, or arrive pre-installed with software. Older machines with unwanted software may also be re-used by installing a fresh system instead.

The first task of a system installer is to boot the target machine. To support unbootable targets such as factory-built machines with uninitialised hard disks, the BIOS must support at least one boot method other than from hard disk. The oldest method is booting from a floppy disk, and although this is widely available it assumes a floppy drive is present. These are being phased out. Floppy disks are slow, unreliable, and offer very limited space

for the system installation software by modern standards. Most machines built since 1997 support booting from CD-ROM by emulating the floppy disk boot sector. If a CD drive is present this is faster and offers more space for both the initial boot software and any further software required. The most sophisticated boot method is network booting. Not all BIOS firmware or network cards support this newer feature. The Pre Execution Environment (PXE) is part of the Wired for Management (WfM) industry standard, and enables most machines purchased since 1998 to boot from the local network.

The installer must access appropriate installation media containing the higher level software to be run after the machine has booted. Typically this will be stored on a local CD-ROM or a network file server. A single compact disc can be used to store a software snapshot, and the capacity of a CD-ROM should suffice for a basic Administration desktop using regular file compression. This static snapshot might be suitable if the software is unlikely to change, or if only a stable base installation is needed for adding additional software to. In general a network installation is more flexible, can be faster, offers greater capacity, and scales better for multiple, parallel installs than sharing install discs between clients.

The system installer transfers the software from the selected media to the target machine's local hard drive, and prepares it for booting. This will involve hardware detection, checking disk capacity and configuring network details.

Some of the possible installation methods are discussed below.

1. **Manual Installation**

   The most basic installation is by a system administrator. Typically software will be provided on compact discs, including a bootable installation disc. Some automatic hints may guide the administrator, but ultimately all customisation is manual. Since package selection, hard disk partitioning, hardware configuration and network details must all be entered manually this process is time consuming and prone to human error. Most Distributions have their own installation program, for example Red Hat's *anaconda* and SuSE's *YAST2*.

2. **Image Cloning**

   If near-identical clones are adequate, a “golden client” can be manually installed and then replicated. Live distributions such as *Knoppix* (which boots a full GNU/Linux environment from a single CD-ROM – see http://www.knopper.net/knoppix/) and other rescue discs can be used to copy the filesystem images of the golden client to other machines. Configuration and customisation can be added by scripts run before or after the installation. Since raw filesystems can be copied to disk rather than the files contained this can offer the fastest possible installation time. However configuring non-identical clones is less efficient and requires expert skill.

3. **Fully Automatic Installation**

   FAI (http://www.informatik.uni-koeln.de/fai/) installs the *Debian* distribution automatically. Software packages are accessed from a *Debian* site, which can be mirrored locally for speed or customisation. The installation kernel provided can be booted from the network or floppy disk, but CD-ROM booting is currently still under progress. Although *FAI* was designed for identical replication of clustered machines, the *cfengine* software described above is used for system configuration and allows extensive flexibility if required.

4. **System Imager**

   System Imager (http://www.systemimager.org/) provides system installation, configuration and maintenance for large networks of machines, preferably with similar hardware, across several distributions. It can boot from floppy disk, CD-ROM or PXE network servers. Both *Debian* and *Red Hat* installations have been tested, but the *System Configurator* software used aims to support all GNU/Linux distributions.

   A golden client is manually installed and configured. Its filesystems are then mirrored to an image server, which target machines install from. If the golden client is updated, these changes are propagated to replicated clients using *rsync*. Although *rsync* sends file differences minimally over the network it can require significant memory to do so. Since modifications are relative to the golden client, *System Imager* is most suitable for target clients with identical or very similar hardware.

5. **Red Hat Kickstart**

   Kickstart (http://www.tldp.org/HOWTO/KickStart-HOWTO.html) is Red Hat's automated installation software. It installs *Red Hat* distributions from CD-ROM, hard disk or network, and boots from network, CD or floppy disk. The *anaconda* installer offers text or graphical interfaces, and can be interactive or fully automated by a configuration file. The hardware detection software *kudzu* caters for a wide range of devices automatically. General installation options can be set up in the configuration file, and extensions added with pre- and post-installation scripts.

   With its intelligent configuration and detection software, *kickstart* can be used to automate similar installations across a variety of hardware targets. Selection of packages from the standard *Red Hat* distribution is straight forward, but updates or extensions can also be included by customising the *kickstart* process.

12.4.3.2. Software Maintenance

Software installations do not remain static during their lifetime. Software updates such as security or bug fixes will be released after the initial installation. In addition, package removal or addition will be required to manage software without re-installing an entire system.

Wherever possible, updates should be done by using “pull” rather than “push” techniques. The decision to pull updates should be made by a machine, either server or desktop, after it has verified itself against a master server. Updates should not be under the control of users. In this way, machines can be kept at the same revision level.

1. **Manual Software Maintenance**

   System administrators can maintain software updates manually. This might involve logging into the target client remotely, copying updated packages, and installing them with the distribution's native package manager. However although this offers tight control to the administrator, it is prone to errors and makes synchronising large collections of machines difficult. Some distributions offer update tools to maintain their standard packages, but typically still require manual intervention and may not provide for extensions to the basic distribution.

2. **Ximian Red Carpet**

   Red Carpet (http://www.ximian.com/products/redcarpet/) is a freely available software-

updating suite from Ximian. It began as a graphical package manager for Ximian's desktop software, but now offers secure remote command line access and more software channels including distribution updates. Mandrake, SuSE and Red Hat are currently supported. It offers easy remote administration and automation, so large numbers of clients can be centrally maintained. However, some hangovers from its original design still remain. It does not support kernel or architecture-optimised updates. It can be configured to update software from customised channels. A proprietary server product *Red Carpet Enterprise* can be used to facilitate management of large software collections.

The graphical interface should not be used as this allows users to control updates. The command line interface should be incorporated into scripts which update the machine automatically.

3. **Red Hat Enterprise Network**

   Red Hat offer a range of software update services as part of their proprietary Enterprise Network (http://www.redhat.com/software/rhen/software_delivery). The most powerful is their *Satellite Server*, which allows full customisation of updates and errata. All servers support their standard *Update Agent* clients for distribution. The same comments against allowing the use of the graphical interface apply here as for *Red Carpet* above.

4. **Debian APT**

   APT is a standard suite of tools supplied with the *Debian GNU/Linux* distribution which allows automated updates to the software installed on a machine. It is able to check dependencies between software packages installed on the machine and available from the software repositories it has been configured to check, and to retrieve and install relevant updates available from a repository. Organisations can set up and maintain their own repositories of software to be installed on their clients (*Debian* includes tools to set up and maintain such repositories), can use repositories provided by Debian and others, or use any combination of these sources of updated software. *APT* has been ported to work on rpm-based operating systems such as *Red Hat Linux* and *Mandrake*, where it provides functionality similar to, and in some ways improved by comparison to, Red Carpet.

12.4.4. Hardware management and system monitoring

Hardware can be monitored for faults and potential faults, for instance by making use of SMART-enabled hard disks and system health-checking hardware. Hardware and software systems should also be monitored for failures, potential failures, absence of service and lack of capacity.

12.4.4.1. MRTG and Snmpd

MRTG (Multi-Router Traffic Grapher, http://people.ee.ethz.ch/~oetiker/webtools/mrtg/) is a monitoring tool originally designed to track and graph the usage of capacity on network links. However, it has developed into a tool capable of tracking virtually any changing quantity, and can be used to monitor such variables as processor, memory and disk space usage, usage of network services including statistics about volumes of email processed, web pages served, etc., system temperature and fan speeds, and other variables.

Snmpd (Simple Network Management Protocol Daemon, http://net-snmp.sourceforge.net/)

is a system management server which can be run on each desktop machine in an organisation. It provides system management information to clients; typically to a central SNMP client which aggregates statistics from a number of machines. *MRTG* can act as such a client and perform this function, providing a graphical overview of the status of a large number of client machines.

12.4.4.2. Nagios

Nagios (formerly known as *NetSaint*, http://www.nagios.org/) is a customisable host, service and network monitoring and management system. It is able to monitor network services and perform various recovery procedures if it discovers that a service is unavailable or having problems, including both invoking automatic recovery scripts and alerting system administrators to the problem. *Nagios* can also provide reports and overviews of the current and past status of the services it monitors.

12.4.4.3. smartd

The *SmartMonTools* toolset (http://smartmontools.sourceforge.net/) includes a daemon called *smartd* which is designed to monitor the SMART (Self-Monitoring, Analysis and Reporting Technology) function of modern hard disk drives. Since these devices are one of the most common components to fail in a modern computer, SMART is intended to monitor drive parameters and warn a system administrator of potential failures before they happen. *smartd* is designed to receive these warnings and take action, typically by alerting a system administrator.

12.4.5. Printer management

12.4.5.1. LPRng

LPRng (http://www.lprng.com/) is an actively developed implementation of the old BSD standard lpr/lpd system. It contains a number of enhancements which make it much more robust and easier to manage than the original products. The author is particularly keen on ensuring that *LPRng* is secure. Until recently this was probably the choice for printer management but recently *CUPS* has made progress and the choice is now less clear cut.

12.4.5.2. Common Unix Printing System

The Common Unix Printing System or *CUPS* (http://www.cups.org/) is designed to be an enterprise-ready Unix print system. It is based on the standard Internet Printing Protocol or IPP, and incorporates a browsing function which allows details of the names and characteristics of printers to be automatically distributed across the network. *CUPS* also incorporates a web-based user interface for administering and configuring the printers. Drivers are available for most common printers.

12.4.5.3. Kprint and GnomePrint

KDE and *Gnome* both incorporate their own printing subsystems which are able to interface user applications with most of the commonly used print spooling systems, including *LPRng* and *CUPS*.

12.5. Backup and recovery

All user and Administration data is assumed to be on one or more servers. It is necessary to be

able to do incremental dumps, find dumps with specified files and restore individual files or whole file systems. Backup of user data tends to easier in Unix and OSS systems then with *Windows* because the user data files including their configuration data are usually contained in a single directory. This is another area where a proprietary product such as *Legato* may be necessary to obtain the features and fine control required in a large site.

12.5.1. Dump and Restore

These two programs are delivered as part of most distributions and are sometimes used together with *tar* and *cpio* in customised scripts to backup and recover single machines.

12.5.2. Amanda

Amanda (See http://www.amanda.org/) is a client server product designed to backup multiple machines to a single device. It is also able to backup *Windows* machine through *Samba*.

12.6. Other services

12.6.1. Time Servers

It is essential in a highly networked environment that all machines – both servers and desktops – have the same notion of the current time. One or more servers are designated as master servers and they get their time either from an attached clock or from external time servers on the Internet. All the other machines are slaves synchronising against these masters.

Synchronising time can be done by running *ntp* (http://www.ntp.org/) on the machines, it can keep a network of machines to within a second of each other quite easily.

Chrony (http://go.to/chrony/) is an alternative to *ntp*. It has some features making it more suitable for higher-stratum NTP nodes than *ntp*, while *ntp* is better for low-stratum nodes which may have to interface directly with such things as GPS receivers and atomic clocks. There are also OSS products for *Windows* which are useful in a mixed environment, such as *Automachron* and *nettime* – http://go.to/chrony/ gives details of both.

12.6.2. Network infrastructure servers

These are the services necessary to run a TCP/IP based network.

12.6.2.1. Routing

Routers allow a large network to be split into smaller interconnected ones. The routers have the job of directing packets from one sub network to another to enable them to get to their eventual destination. Building routers requires a good understanding of the basic protocols and many Administrations will probably want to purchase proprietary dedicated routers.

However for those who want to build their own two products exist *Bird* (http://bird.network.cz/) and *GNU Zebra* (http://www.zebra.org/).

12.6.2.2. DNS

A TCP/IP network needs some means of translating IP addresses into human meaningful domain names and vice versa. DNS is a protocol together with a number of inter communicating servers each with data on them. DNS is basic to the working of the

Internet. There are a number of programs to build DNS servers including *BIND* (http://www.isc.org/products/BIND/), *MyDNS* (http://mydns.bboy.net/), and *MaraDNS* (http://www.maradns.org/). *BIND* is the most widely used.

12.6.2.3. DHCP

DHCP is a protocol described in http://www.dhcp.org/ which allows machines to obtain their network details at boot time from a central server or servers. DHCP allows the efficient use of scarce IP addresses and will reallocate addresses where possible.. It also allows central administration of many global addresses such as gateways and name servers. The main product comes from http://www.isc.org/products/DHCP/ and consists of a client and server application. The client has to run on all the participating client machines. Both these products come with most standard distributions.

12.6.3. File servers

Network file servers allow network attached machines to access file storage on a remote machine as though it were local.

12.6.3.1. NFS

This is the de facto standard and has been in use for many years. The commonly implemented subset does not provide strong security, though a secure variant is defined and is implemented in some commercial Unix variants.

NFS consists of a server which exports files from the machine it is running on to clients running on other network attached machines. There are controls over which other machines can attach to these files but once one is attached then traffic over the network is in the clear. There is minimal authentication of attaching users in the Linux version.

The other problem with any networked file system is that a network failure will stop file access. To overcome this a distributed file system is required (see below).

NFS is a standard part of most distributions.

12.6.3.2. Samba

Samba is a product which implements Microsoft's SMB protocol. See 14.5.1 for a more detailed description. It is critical to the integration of OSS and Windows based systems and comes with most standard distributions. Its use is described in some detail in Chapter 14.

12.6.3.3. Netatalk

For those who have Apple Macintosh machines *netatalk* provides the implementation of the AppleTalk protocol. (See http://netatalk.sourceforge.net/).

12.6.3.4. OpenAFS, CODA and Intermezzo

These products implement a distributed file system to varying degrees. With such a system access to files can continue when the network fails because local caching provides the appearance of being connected. This is a non trivial problem and the products solve it in different ways. This sort of filesystem is really needed with laptops or machines attached to a transient connection. The other way of providing the same functionality is to have local storage which is synchronised with a central server periodically. See

http://www.openafs.org/, http://www.coda.cs.cmu.edu/, and http://www.inter-mezzo.org/ for details on each product.

http://www.inter-mezzo.org/docs/bottlenecks.pdf contains a detailed discussion of the characteristics of all the above.

12.6.4. Directory services

The provision of rapid lookup of names and address and associated data.

The most popular standard for directory services is LDAP. This is an open protocol and is implemented in many products for instance *Evolution* and *OpenOffice.org*. LDAP works with data definitions called schemas and it is possible for Administrations to develop their own customised schemas. Unfortunately the schemas used by applications are not always compatible with one another, which means, for instance, that is difficult for *OpenOffice.org* to read *Evolution* data and vice versa.

The OSS application *OpenLDAP* conforms to the LDAPv3 standard, and version 2.1 and later can be configured with a range of different database back ends (such as flat file, SQL or even site defined).

Most of the groupware suites provide some form of directory service, but few will actually integrate compatibly with LDAP. Short of using cut-and-paste techniques, it is difficult to use the contact database they provide in external mail agents. Most of them offer mail agents of their own, but are not very forthcoming about the level of integration available to the inbuilt contact manager.

OpenOffice.org, *Evolution* and *Mozilla* provide integral address book functions. However, the storage formats used are not interchangeable. To allow interworking some site tailoring is needed.

12.6.5. Legacy support

12.6.5.1. Terminal emulation

The use of *xterm* with an appropriate TERM environment variable setting can emulate most character-based terminal types, for instance VT220 and VT100. There is a specific 3270 emulation named *x3270*. Page-based emulations can be found in proprietary products.

12.6.5.2. Remote display

There is a discussion of this in Section 13.3 below.

12.6.5.3. Emulation

There is a discussion of this in Section 13.4 below.

13. Application Migration - Overview

Once a list of applications has been drawn up it can be grouped into the following categories:

13.1. Proprietary applications which have an OSS equivalent

Some applications, for instance *Office*, *Lotus SmartSuite*, *WordPerfect*, *Framemaker*, *Quark Express* and *Photoshop*, have equivalents which run natively under OSS, including *OpenOffice.org*, *Gnumeric*, *Evolution* and *The GIMP*. In this case the OSS product needs to be tested to ensure it provides the necessary functionality.

13.2. Proprietary applications which run in an OSS environment

Some applications, such as Adobe's *Acrobat Reader*, have a version which runs natively under OSS. If there is no OSS alternative to the application then all that is required is to ensure that all the required features are implemented in the proprietary version. If there is an OSS alternative and a partial migration is acceptable, then a choice has to be made based on the features offered by the proprietary and OSS applications.

13.3. Software which may be accessed by remote display

Another approach is to run the applications on a server and transport the display to the desktop; this is the thin client approach. Products like *Windows Terminal Server*, *Citrix* and *Graphon* allow applications to run on a server running *Windows* in a multi-user way. This means that an application written to run on the desktop in a single user mode may have to be altered to run under these products. This will not be possible without source code, and third party vendors may not be willing to help.

The most sophisticated of these products, *Citrix*, has its own line protocol, “ICA”, which is extremely good, particularly with low bandwidth connections. It can run a server farm with load balancing and has other useful facilities. There are free ICA clients which run under GNU/Linux.

All these products rely on proprietary closed source software and *Citrix* in particular is expensive. It requires a *Windows* server licence, a *Citrix* licence and a *Windows Terminal Server* licence if a non-*Windows* client is used. In addition, Client Access Licenses will be needed for each desktop using the software. The *Citrix* licence is based on concurrent users, so this approach can be cost effective if there are many users who need access to an application but where concurrent access is low. There are documented case studies at http://www.citrix.com/press/news/profiles/ which show that the savings of having thin client “disposable” desktops are sufficient to justify moving the applications to a server. *Citrix* also has products to allow Unix applications to be transported in the same way over ICA and displayed on a thin client desktop.

Windows Terminal Server provides similar functionality to *Citrix* except that it uses its own protocol, RDP. The GNU/Linux client for RDP, *Rdesktop* is good, but is still considered beta code by some. RDP used to be very inefficient in comparison to ICA but the difference is now small if not negligible. *Citrix* has a number of features like load balancing which make it the better choice for large scale installations where the extra cost can be justified.

Both *Citrix* and *Windows Terminal Server* can introduce latency into the application if the servers are not sized correctly and the network is not sufficiently fast.

Tarantella (http://www.tarantella.com/) sits on a server between the desktop and the application servers. It aggregates output from *Citrix* on *Windows* and other applications

running on Unix and IBM mainframes, and sends the result to a browser on the desktop. It uses its own proprietary line protocol, AIP, which is apparently reasonable at low bandwidths. However, it increases latency because it sits between the user and the application and therefore slows down the connection between the two.

As mentioned above, CodeWeavers now produce a server version of their *CrossOver Office* product. This works by having the client securely connect to the central server and have an X session displayed back to it. This means that the communication to the central server is encrypted and compressed but also requires sufficient bandwidth to support it as it is based on X. No tests of the bandwidth requirement have been made but it is likely to be greater than for ICA (*Citrix*) or AIP (*Tarantella*).

VNC is an OSS product developed by AT&T which is designed to display a user session running on another machine. It consists of a server and client which are both available for *Windows*, Unix and GNU/Linux. *VNC* allows applications to be run in one environment and the display to be run in another. It uses its own open protocol, RFB, over TCP/IP, which is not as efficient as either ICA (*Citrix*) or AIP (*Tarantella*) and so needs high network bandwidths (such as 100Mb/s) to work well. Unfortunately the *Windows VNC* server is also not as efficient as the Unix version and can require more processing power than one would expect. *VNC* can be very useful for occasional systems administration use, allowing control of a desktop to be taken by a central person. In these circumstances, high latency could be acceptable.

13.4. Software which will run under an emulator

If none of the above provides a way to run the application or a substitute then it may be possible to run it natively, but with its normal operating environment being emulated on top of an OSS operating system. A good discussion of the issues relating to this approach can be found at http://www.linuxmednews.com/linuxmednews/967526746/index_html. All of these techniques have licence implications because they may involve running multiple copies of the proprietary application and/or operating system.

This section is most likely to apply to *Windows* applications but because the techniques can apply to other Scenarios they are discussed here rather than in Chapter 14.

There are two types of emulation:

13.4.1. Hardware emulation

Products like *VMware* and *Win4lin* provide hardware emulation. They allow a normal PC operating system to run as a user-level application by mimicking Intel PC hardware in software interfaces and thereby providing a virtual machine. This allows a legacy operating system and its applications to be run on top of an OSS platform.

VMware is not strictly an emulator – it allows most instructions to pass straight through to the processor, which means it will only run on an x86 architecture machine. It is the most complete offering but it is proprietary and can consume a lot of machine resources.

Win4lin is similar to *VMware*, and is also a proprietary product, but is less expensive. It can be a good solution in simple cases – for example just running office applications. It is a component of the *Lindows* product which is being sold on low cost hardware to home users. (Because it apparently does not use unprivileged user accounts to maintain security, *Lindows* itself should not be recommended for Administrations without careful consideration of the security implications.)

Because the hardware emulation approach requires full licences for the proprietary operating

system and application, together with the cost of the emulator, it should be viewed as a way of running a small number of legacy applications which are difficult to migrate.

There are *VMware* and *Win4lin* server products which can reduce licensing costs if the proprietary software allows ***concurrent*** rather than ***potential user licensing***.

There are OSS applications which will fully emulate an Intel IA-32 environment, for instance *Bochs*, but they are probably not ready for Administration use yet.

13.4.2. Software emulation

Software emulation allows programs written for a proprietary environment to run directly on the OSS operating system. Any system calls made by them are mapped onto the equivalent OSS system interface. This means that a copy of the proprietary operating system is no longer necessary.

Wine allows applications written for *Windows* to run on GNU/Linux by providing software emulation. *Wine* is described in detail in Appendix B. The major problem *Wine* has to solve is the large number of *Windows* system calls (including bugs) that it has to support.

OSS *Wine* code is available from http://www.winehq.org/ or from CodeWeavers at http://www.codeweavers.com/technology/wine/download.php.

CodeWeavers produce two proprietary products, *CrossOver Office* and *CrossOver Plugin*, which are based on *Wine* and designed to support specific *Windows* applications. Although the products are proprietary, code modifications are periodically contributed back to the OSS version of *Wine*.

CrossOver Office is designed to allow applications such as *Office* and Lotus *Notes* to run natively on GNU/Linux. There are some outstanding issues but the product is in active development. However, this approach may be appropriate for certain users depending on their requirements. *CrossOver Office* is now also available as a server product which means that it doesn't need to be fully installed on the desktop and can provide similar functionality to *Citrix.*

CrossOver Plugin is designed to allow browser plugins that normally only run on *Windows* to run in *Netscape, Mozilla* and *Galeon* on GNU/Linux. This product has been available for longer than *CrossOver Office* and works very well.

Using these techniques removes the cost of the *Windows* operating system licence but not the application licence. The application licence needs to be scrutinised to make sure it doesn't forbid running the application without *Windows*. This restriction is used in some new Microsoft applications as a lock-in tactic, although legal enforcement is questionable.

13.5. Software which can be recompiled under OSS

For applications written in-house or on behalf of the Administration and for which source code is available, the software can be ported to run on an OSS platform. In general, the problem of porting source code in any language is not compilation but the code's usage of system libraries including both the graphical environment and operating system. This can mean a significant amount of manual intervention to migrate the code. In addition, any assumptions about the underlying environment, such as file naming, will make it necessary to either change the source code or replicate the environment, regardless of the language used.

1. **Java.** If Java software has been written according to the Java specification then the program should run without any problems. However, if any proprietary extensions have been used

then the code will have to be changed to use standard modules instead.

2. ***Visual Basic.*** A proprietary product called *DeLux* (http://www.deluxsoftware.com/) can be used to convert *Visual Basic* code to *Kylix* (see item 4 below) and can be run under GNU/Linux natively. **netproject** have not been able to test this product. Microsoft's development tools can convert *Visual Basic* code to ***.NET*** and produce ***CIL*** code. The Mono OSS project allows this code to be run under GNU/Linux. Mono is currently being developed very rapidly and any given application may or may not work depending on the way it interacts with libraries such as the screen display.
3. **C#.** This is increasingly supported under GNU/Linux, and Ximian have produced a compiler as part of the Mono project, adding C# bindings to crucial components of the Gnome Desktop. The Mono project includes an interpreter that allows CIL code produced by proprietary development tools to be run on GNU/Linux unchanged. The Mono project and the use of the .NET development framework is a very lively area of OSS at the moment and the position changes very rapidly.
4. **Pascal and *Delphi*.** Pascal as a free-standing language is little used these days, but it is the essential coding component of Borland's *Delphi* rapid development tool. Borland have a native GNU/Linux equivalent of *Delphi* that goes by the name of *Kylix*. *Kylix 2* and *Delphi 6* are stated to use compatible code syntax and have identical support environments.
5. **C and C++.** Programs written to ANSI standards should recompile and run as long as the underlying system libraries used are compatible. For instance programs written specifically for *Windows* will not in general compile and run correctly under GNU/Linux due to the very different set of calls to both the operating system and run-time libraries such as the windowing system. This mismatch can often be dealt with by compiling the code with *Winelib*, a part of the *Wine* project.

14. Scenario 1 – Windows

The Administration has one or more interconnected *Windows* Workgroups, *Windows NT* PDC/BDC or *Windows 2000* Active Directory domains. All users have *Windows* desktops. All central applications run on *Windows* servers.

Throughout this chapter the word *Windows* means a version of *Microsoft Windows*. Where the precise version is important it will be stated. Code examples are based on a *Red Hat Linux* system; other distributions may have subtle differences.

The content of this Scenario should be read in conjunction with the general comments made in previous Chapters.

14.1. Planning the migration

To recap on what was said in Chapter 5, planning for the transition phase is very important, the success of an OSS project will be judged as much on the smoothness of the transition as on the quality of the final service. It is likely that any practical transition from one system to another will take place over a period of months or even years. During this time, data must be moved, people trained, software installed, and the business of the Administration must carry on without disruption.

Careful planning will be required, and large administrations should run a pilot phase to test the plan before putting it into effect on a large scale.

14.2. Domains

This Scenario can be divided into the following:

14.2.1. Windows “workgroup” model

A group of *Windows* computers co-operating loosely on the network by declaring themselves to be part of the same “workgroup”. There is no security aspect to workgroups – they serve only to group machines conveniently in browser lists.

Users wishing to share files with others can make available “shares” – parts of their directory hierarchy – either for general access or with a password being required.

There is no co-ordination of usernames and passwords in this model. Indeed, with some versions of *Windows* there is no real concept of file ownership.

Migrating from a workgroup scheme to another will involve collecting up important files by hand, one machine at a time.

14.2.2. Windows NT domain

In this model, one or more computers act as domain controllers to co-ordinate usernames and passwords. One of these server machines is designated the Primary Domain Controller or PDC, and all changes are handled by that machine. There may also be one or more Backup Domain Controllers or BDCs to provide redundancy and load-sharing.

Windows NT domains usually include one or more fileservers (which may be the same machines that are running PDC and BDC functions). The fileservers provide storage for roving profiles (user desktops, documents, and settings) and may also provide “home directory” space, shared filestore, and print spooling services.

In a well-managed domain, users are normally told to keep all files in their desktop or their home directory so that no important data is held on individual PCs. Migrating data from such well-managed environments to new systems is relatively simple, as the system administrators know where to find all the files that matter.

14.2.3. Windows 2000 Active Directory domain

The *Windows NT* domain model becomes very hard to manage effectively for large numbers of users, so *Windows 2000* introduced a hierarchical domain model. This is known as Active Directory or AD and it makes use of ideas from both the Internet Domain Name System (DNS) and the Lightweight Directory Access Protocol (LDAP).

As in *Windows NT* domains, AD usually provides fileservers to hold roving profiles and home directories so it should be simple to find the important files when planning the migration process.

Because AD allows LDAP access, there are more migration options available to a site that uses AD. For example, it should be possible to use the AD servers to hold username and password data for OSS servers and clients: this could be convenient where a small part of the total user base is to be moved to OSS, as the user management process can be left almost unchanged.

14.3. Overview of possible migration routes

The two main routes considered here are:

1. Add OSS machines to existing *Windows* domains and gradually move data and users across, followed by removing the old proprietary servers. It is possible to migrate clients and servers independently.

   Adding servers to the *Windows* domain is one of the fastest ways to gain benefit from OSS. For instance the combination of the GNU/Linux Operating System with *Samba* gives a powerful and low-cost file/print server that can be used in place of an *Windows* system without any changes to the existing client environment.

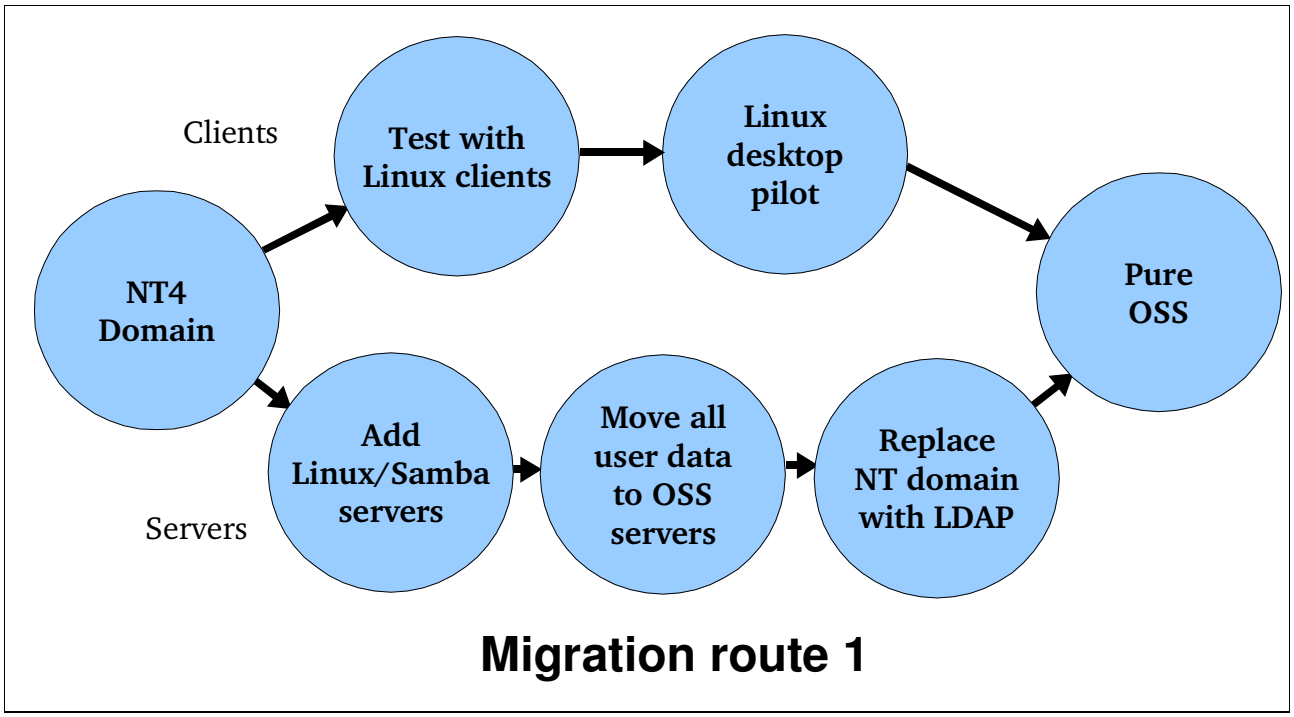
   Running OSS clients in a *Windows* domain is a low-risk form of co-existence, as no changes to servers are required. It can be used where a small number of people will be using OSS desktops in an otherwise *Windows*-only environment.

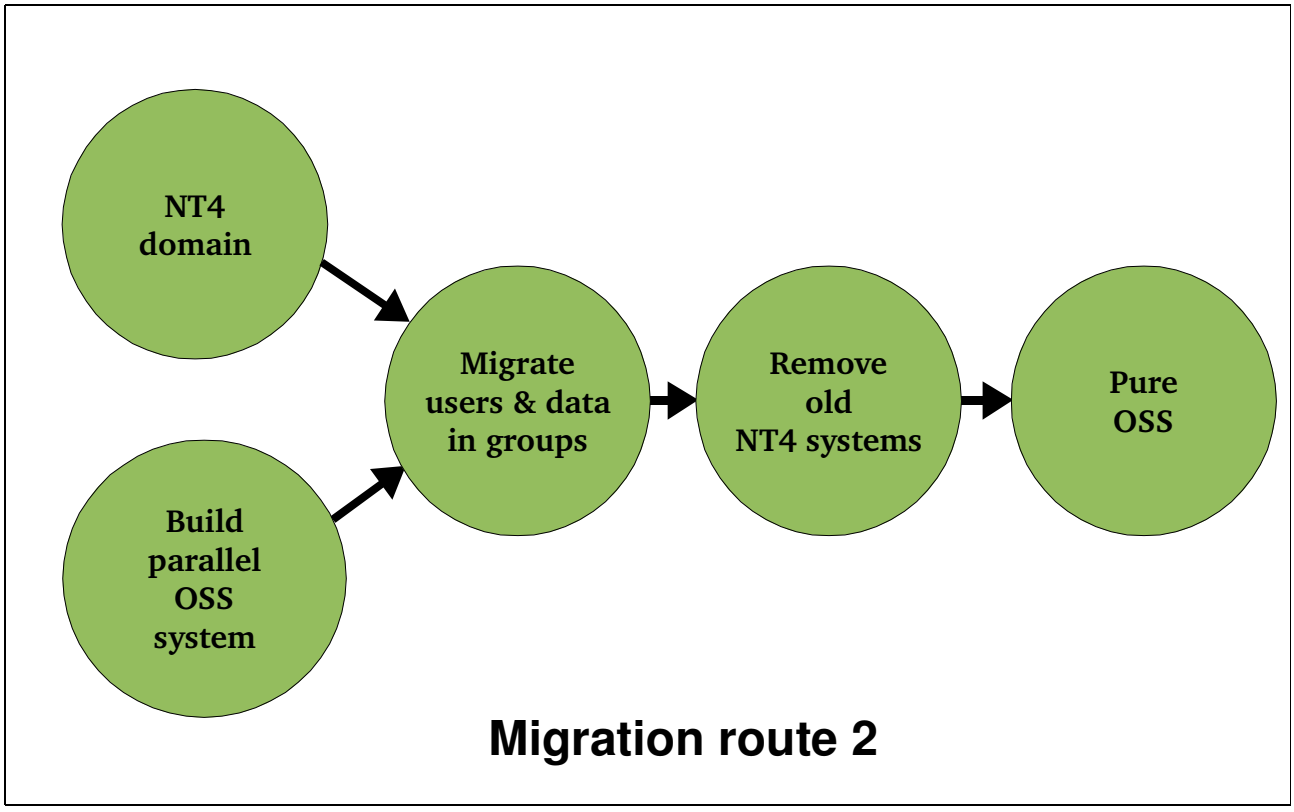
2. Build a parallel OSS-based infrastructure and migrate users and their data in groups, with minimal interaction between old and new systems.

   This is much simpler than running a mixed *Windows*/OSS system, but it does make co-operation between people using *Windows* and those using OSS systems rather more difficult.

Both routes are summarised in diagrams below. The first route provides tighter integration between old and new systems during the transition, but requires significantly more planning and implementation effort.

A constraint on the choice of route will be the way in which the Administration is organised and how this maps onto the logical and physical structure of the computing installation.





The early stages of most migration routes include a co-existence phase, where both `Windows` and OSS systems are in use often accessing the same data. These can be particularly useful models where a partial migration is planned, with some groups moving to OSS and others staying with the old system.

The technical details of doing these changes is in Section 14.6 onwards. But first we discuss the technical background and the tools needed.

14.4. General issues

There are many similarities between current proprietary systems and the Open Source systems that could be chosen to replace them. In particular, graphical user interfaces (GUIs) have tended to converge on a fairly standard "look and feel" which reduces problems for end-users moving from one system to another. End-user training will still be required, to help people deal with the things that are different and to get the best out of the new system.

Behind the similar appearance of the GUIs, there are some important differences between `Windows` and OSS systems. These are particularly apparent at the system administration level. It is here that most training and planning will be needed. OSS systems such as GNU/Linux do have management GUIs, but large installations are more commonly managed using command-line tools, as these lend themselves to scripting, process automation, remote management and advanced control. It is this ability to automate tasks that makes Unix and OSS system administrators so productive.

In addition to the differences in management processes, there are also some important differences in the service provided. These must be planned for and dealt with during the transition.

14.4.1. Usernames and passwords

Computer users identify themselves using usernames and passwords. In some Administrations they may also use smart cards or other cryptographic devices to provide stronger proof of identity.

14.4.1.1. Username issues

Some Administrations may have "structured" usernames that encode information about the user. For example, the username *cfg27* might belong to the 27th person to be registered in the Corporate Finance Group. Others allow people to choose their own usernames, or simply use their real name. Structured username schemes can normally be used in OSS systems without change. OSS usernames cannot start with a numeric, which can cause difficulty with structured usernames where the initial structure could well be numeric.

There are some issues that might affect the more ad-hoc systems. Usernames in `Windows` systems are generally case-preserving and case-insensitive. This means that if a person is given the username "Mary", she can type "mary", "MARY", or even "mArY" at login time without problems. It also means that whenever the system displays a username (such as as the owner of a file) it will use the form originally typed by the administrator when the username was created – in this case, "Mary".

On the other hand, usernames in Unix and OSS are case-sensitive. The user must type their username exactly in the form that was originally registered. Conventionally, usernames are made up entirely of lower-case letters and numbers with no other characters used, and with a maximum length of eight characters.

These restrictions have been greatly relaxed in recent years, and modern systems will permit much longer usernames with a wider character set. Certain authentication and authorisation schemes now implement case-insensitive usernames: the LDAP-based scheme proposed in this document is such a scheme, so usernames like "Mary" and "FinancialController" are quite possible. Care should be taken though, as there may be older packages around that make assumptions based on the old rules. In particular, it

would be very unwise to allow spaces or certain other punctuation characters in usernames.

It would be good practice to limit usernames to use those characters allowed in mail names so that login usernames can be used as mail names as well.

14.4.1.2. Password issues

Modern OSS systems allow passwords of almost any length, and permit a very wide set of characters to be used. It is good practice to encourage the use of long passwords (10 or more characters) with a good mix of letters, numbers, punctuation and varied capitalisation. The password setting utilities generally refuse to set very weak passwords unless forced by an administrator, and many sites may decide to enforce even stronger rules.

Some commercial Unix variants still truncate passwords to eight characters so if a mixed environment is planned, this should be taken into account.

Migrating passwords from existing proprietary systems to new OSS systems is not always possible, as passwords are usually held in an encrypted hashed form. The transition plan may have to include the reissue of passwords to all users, or possibly a password collection and synchronisation phase.

14.4.2. Authentication services

Any network of more than a handful of computers needs a networked naming and authentication service. In *Windows NT* this is known as the Domain Controller. In later *Windows* systems it is Active Directory. Novell NDS is also widely installed, and other proprietary systems have their own naming and authentication systems.

Most Unix and OSS systems can interwork with almost all common naming and authentication services. GNU/Linux is particularly strong in this respect. The service proposed in this document is based on LDAP, but it is also possible to use multiple naming and authentication systems at once, which may be useful during the transition phase.

14.4.3. Files

A very important part of any transition plan concerns the migration of data from the old system to the new one. If a “big bang” migration is planned then this will be a one-off operation, but if in the more likely instance parallel running is envisaged, then cross-platform file access will be necessary. Great care must be taken to avoid data loss, and to avoid the confusion that could result from having separate modifiable copies of a file in both “old” and “new” environments.

14.4.3.1. Content and format

This is the most obvious migration issue, and is dealt with in detail in Section 14.8 below. The normal approach is to use OSS applications that can read files written by the proprietary application that they replace, though in some cases it may be appropriate to plan for a bulk format conversion as part of the migration process.

Special data such as macros and scripts are likely to need attention from skilled programmers during the migration.

14.4.3.2. File names

As with usernames, *Windows* filenames are case-insensitive and (to some extent) case-preserving. Some *Windows* applications do tend to capitalise the first letter of filenames and do other changes that the user may or may not be aware of. The *Windows* environment also carries the legacy of the DOS “8.3” filename format, which still shows up in some utilities. *Windows* filenames commonly contain spaces, and normally use the Unicode character set. *Windows* uses “\” as the directory separator.

Although it is less obvious to GUI users, *Windows* absolute filenames must include a “drive letter” indicating the physical device that holds the file, or they must have the actual name of the server if the file is on a “network drive”. These restrictions can be a problem to managers of large *Windows* systems who try to provide a seamless service in the face of hardware changes.

Other proprietary systems treat filenames in different ways. *VMS*, for example, has case-insensitive filenames that usually include one dot and may include a version number after a semicolon.

Filenames in Unix and OSS systems have different rules. Here, filenames are fully case-sensitive and the system does not make any changes to names supplied by the user. Names use an 8-bit character set determined by the locale currently in use (in most of Europe, the character set is ISO 8859-15). The only characters GNU/Linux does not permit in filenames are the directory separator (“/”) and the null character. However in practice it is unwise to include non-printing characters, for example the *Windows* FAT32 filesystem cannot store the first 32 ASCII codes or any of ", *, :, <, >, ?, \ or |. Spaces in filenames are permitted, though their presence does require command-line users to be more careful with quoting.

Unix and OSS systems do not use drive letters, and do not require the real name of the fileserver to be part of the absolute filename where network file access is used. Instead, the system presents all files as part of one seamless hierarchy. Together with the use of symbolic links in the filesystem and data-driven automounters, this gives system administrators great flexibility in separating the absolute name of a file from its physical storage location.

Almost all *Windows* filenames can be migrated directly to OSS servers without change. The only exception likely to be encountered in practice is filenames containing the “/” character, which will need to be modified during the transition. Users of GUI tools will probably never notice that filenames have become case-sensitive as they only ever type such names when first creating the file.

14.4.3.3. Dual access

Many migration plans are likely to include a period of parallel running where some people are using OSS systems and others are still using proprietary ones. Where files need to be accessed by members of both groups, special provisions may be needed.

File sharing in *Windows* systems uses the SMB (Server Message Block) protocol, which is a very complex technology with multiple levels of backwards compatibility. It is used with dedicated fileservers and also in “peer-to-peer” mode, where individual PCs make available parts of their filesystem on the network. Well-managed Administration environments are likely to be based on dedicated servers rather than ad-hoc sharing.

Non-shared user files in a *Windows* environment may be held in several different places:

1. On a local disk of the user's desktop or laptop PC – for instance the one referred to as "the C drive".
2. In the user's "roving/roaming profile" – this includes most preference settings and also the content of the `Windows` desktop and (usually) the "My Documents" folder. The roving profile is held locally on whatever PC the user is actively using, and is synchronised back to a profile store at logout time. This provides a handy backup facility, but can have serious performance implications with users reporting very slow logouts.
3. In a "home directory" on a centrally-managed fileserver. This is a common option for large networks of desktop systems as it is easy to manage backups properly.

It is not sensible to try to provide dual access to files held on individual desktop or laptop PCs, so any files on local disks or in roving profiles should be moved to managed fileservers at an early stage in the migration process.

The main networked file access mechanism for Unix and OSS systems is the Network File System (NFS). This is a much simpler protocol than SMB, and its specification has always been openly available.

Options for implementing dual access fall into two broad categories: adding dual-protocol support to the servers, or adding dual protocol support to the clients. Other things being equal, it is normally easier to change servers than clients, and almost always easier to adjust OSS systems than proprietary ones. The options are summarised in the table:

	***Windows* servers**	***OSS or Unix servers***
***Windows* clients**	SMB file access is standard.	Servers support SMB using the Samba package. This is mature software with excellent performance.
OSS Clients	GNU/Linux clients can access SMB shares. This is fairly easy where client machines have just one user at a time, but gets more involved where time-sharing machines are used. Commercial Unix variants do not normally have SMB client capabilities. It is possible to add NFS service to `Windows` servers, but this can be very expensive.	NFS file access is standard. GNU/Linux clients can use SMB if desired as part of a migration plan, but this is less efficient.

14.5. Tools

This section discusses some of the key OSS components that will be used when migrating from proprietary systems.

14.5.1. Samba

Samba is a file and print server package for OSS systems. It implements Microsoft's SMB protocol and in many cases can completely replace the functions of a *Windows* server. *Samba* can also act as an *Windows NT* Domain Controller and is capable of storing domain management data in a directory accessed using LDAP.

Samba also provides SMB client tools suitable for scripting, which are very useful when diagnosing problems with SMB networks and also when doing bulk file migration from *Windows* servers.

Samba is maintained by a core group of about 30 very active volunteers around the world. More information can be found at http://www.samba.org/.

14.5.2. OpenLDAP

OpenLDAP is an implementation of the Lightweight Directory Access Protocol (LDAP). It includes a directory server, a set of data access and management tools, and a set of libraries to support LDAP in other applications.

OpenLDAP is maintained by a small core group plus a large number of contributors. One of the core group works on the project full time.

14.5.3. NSS and PAM

NSS is the Name Service Switch: a technology used by GNU/Linux and some commercial Unix variants to allow different name services to be used when looking up hostnames, usernames, groupnames etc. Many modules are available, of which the ones most relevant to this project are:

1. files: Simple lookups based on local text files
2. DNS: hostname lookups based on the Domain Name System
3. LDAP: lookups based on LDAP – mostly for usernames and groupnames but also usable for many other purposes.
4. SMB: lookups using *Windows*' SMB protocol (See 14.5.5 below)

PAM is the Pluggable Authentication Module system. Like NSS, it is common to GNU/Linux and several commercial Unix derivatives. PAM is also available on *FreeBSD*. PAM allows great flexibility in configuring the authentication and authorisation process. Relevant modules include:

1. LDAP: uses LDAP *bind* operations to verify user credentials.
2. SMB: Uses *Windows NT* domain operations to verify user credentials.
3. Access: restrict access to networked services.
4. Cracklib: enforce quality checks on new passwords.

14.5.4. GNU/Linux SMBFS file access

Samba allows an OSS system to provide a file service to *Windows* clients. SMBFS works the other way around: it allows an OSS system to access files held on *Windows* servers. SMBFS is provided with most recent GNU/Linux distributions but is not normally found in commercial Unix systems.

The access-control model used by *Windows* filesystems is different from that used by GNU/Linux and other OSS systems, so there are some limitations in what can be achieved

with SMBFS.

14.5.5. Winbind

Another product from the Samba team, *Winbind* allows individual GNU/Linux machines to be attached to *Windows NT* domains. It maintains a mapping between *Windows NT* authenticators (SIDs) and Unix-style UIDs and GIDs. *Winbind* can do many other things that reduce the load on system administrators, such as setting up Unix-style environments for people when they first log in.

The disadvantage of *Winbind* in large networks is that each client computer builds its own mapping between *Windows* authenticators and Unix ones. This can cause problems in the later stages of migration when OSS fileservers are introduced.

When using *Winbind*, usernames and group names used by GNU/Linux are formed by concatenating the *Windows NT* domain name with the *Windows NT* username to form a unique string. This can lead to some confusion, as many common Unix-style utilities only provide space in their output for eight-character usernames. The longer names generated by *Winbind* get truncated in the display.

14.6. Migrating the operating system environment

14.6.1. Add individual GNU/Linux servers to an existing Windows NT domain

Setup is extremely simple:

1. Install a GNU/Linux server, giving it a fixed IP address.
2. Make sure that the *Samba* packages are installed, typically *samba*, *samba-common* and *samba-client* are required. These are normally included in a "server" installation.
3. Edit **/etc/samba/smb.conf**, set **domain** security mode, and define the domain (workgroup) name. List the PDC and any BDCs as password servers. Define the shares that are to be served by the machine.
4. Create any directories that are to be shared, and set appropriate ownership and permissions.
5. Join the machine to the existing *Windows NT* domain using the Domain Admin password (or any other username and password that has the power to do this):

```
smbpasswd -j DOMAINNAME -r PDCNAME -U Administrator
```

6. Start *samba* and arrange for it to restart at reboot:

```
/etc/init.d/smb start
chkconfig smb on
```

The server will now show up in browse lists and can be used just like an *Windows NT* server.

14.6.2. Run GNU/Linux desktops in Windows NT domains

14.6.2.1. Simple setup for small numbers of machines

In the early stages of testing OSS tools it is very useful to run individual GNU/Linux machines with very simple configurations. These can be given access to files on *Windows* servers for compatibility and migration tests using the **smbmount** command.

Mounting is the Unix/OSS term for making a disk or remote filesystem part of the local machine's file hierarchy. The process is usually done automatically at boot time under the control of the **/etc/fstab** file, but can also be done interactively. For example, the command to bring an ISO-standard CDROM into the filesystem under **/mnt/cdrom** would be:

```
mount /dev/cdrom /mnt/cdrom
```

The **mount** command is normally restricted to use by the *root* user for security reasons. This is not a problem where the machine is being used by a system administrator, but can be awkward where a non-technical user is involved. GNU/Linux provides several ways around this problem:

1. Use a special entry in **/etc/fstab** that allows ordinary users to mount certain pre-defined objects. This is the usual way to allow CDROMs and floppy disks to be mounted on demand. Files on the mounted device usually show up as being owned by whoever caused the device to be mounted.
2. Use a **setuid-*root*** program to do the privileged operation, having first checked that it is safe. This is the easiest way to handle the mounting of remote *Windows* shares.
3. Use an **automounter** to mount filesystems when they are first accessed and to unmount them when no longer in use. The automounter runs as a daemon and is usually driven by network-wide configuration data. This takes more effort to set up than the other methods, but is extremely useful in large networks.

In this scheme, we will use the **smbmount** and **smbumount** commands to make an existing *Windows* share appear as part of the local GNU/Linux filesystem. On *Red Hat Linux* systems these are part of the *samba-client* package, so make sure you have installed both the *samba-common* and *samba-client* packages. These programs are designed so that critical parts can be given *root* privileges, although they are not normally installed that way by default so a few commands must be run by *root* before they are first used:

```
chmod u+s /usr/bin/smbmnt /usr/bin/smbumount
```

Note that the command changes **smbmnt** rather than **smbmount**. This is important because **smbmnt** encapsulates only the functions of **smbmount** which require *root* privileges. Having done this, any user can use **smbmount** and **smbumount** and they will run with the necessary *root* privileges.

Now, any user can make a *Windows* SMB share available as part of the GNU/Linux filesystem by mounting it on a directory that they already own. Any files that are already in the directory will be invisible while the SMB share is mounted on top.

As an example, suppose the GNU/Linux user *fred* wants to access files on a *Windows NT* server called NT4SERVER in the domain THESTATE, which are shared under the sharename FREDERICK and owned by the *Windows* user FREDERICK. *fred* starts by making a new directory to mount the *Windows* share onto:

```
mkdir ~/ntfiles
```

(The notation "~/" means "in my home directory".) This only needs to be done once. Now, to mount the remote share:

```
smbmount //nt4server/frederick ~/ntfiles \
```

```
-o username=frederick -o workgroup=thestate
```

The command should be typed on one line, or split up with "\" line-continuation characters as shown here. It will prompt for FREDERICK's password on the server, and then mount the *Windows* share on top of the directory *ntfiles* in *fred*'s home directory. To avoid typing the whole thing at every login, it can be put into a script file or even made part of *fred*'s login process.

The mounted share now behaves almost as if it were part of the local disk. Files can be created, deleted, and edited. There are some caveats though. In particular, there is no attempt to map between Unix-style access control and *Windows NT* ACLs so commands to change the ownership or mode of files and directories on the mounted share will have no effect.

Before logging out, it would be wise to unmount the share:

```
smbumount ~/ntfiles
```

Again, this could be made an automatic part of the logout process if required.

The process described in this section does not create any permanent link between accounts on GNU/Linux and accounts on existing *Windows NT* servers, so usernames and passwords must be maintained separately on each machine. The management effort involved can quickly become excessive as the number of machines grows, so this scheme is really only suitable for small test environments.

14.6.2.2. Smarter setup for larger rollouts

Where a larger pilot deployment of OSS desktop systems is required, it may still be convenient to keep file and authentication services on existing *Windows NT* servers. *Samba*'s Winbind daemon provides an easy way to link the two environments.

Samba and *Winbind* are standard parts of the *Red Hat Linux* distribution, but they may not be installed by default on workstation setups. To use *Winbind*, the following packages should be installed: *samba, samba-common* and *samba-client*.

The file **/etc/samba/smb.conf** must be edited to show the correct *Windows NT* domain name in the **workgroup** line, and to put the system into **domain** security mode. *Winbind* configuration data also goes into the global section of this file, for example:

```
# separate domain and username with '+', like DOMAIN+username
winbind separator = +
# use uids from 10000 to 20000 for domain users
winbind uid = 10000-20000
# use gids from 10000 to 20000 for domain groups
winbind gid = 10000-20000
# allow enumeration of winbind users and groups
winbind enum users = yes
winbind enum groups = yes
# give winbind users a home directory location
template homedir = /home/winnt/%D/%U
# and a shell
template shell = /bin/bash
```

For Winbind to work, certain services need to be running. To start them and to ensure that they start at each reboot, the commands are:

```
chkconfig smb on
chkconfig winbind on
/etc/init.d/smb start
```

```
/etc/init.d/winbind start
```

The machine must now be joined to the *Windows NT* domain. This requires a *Windows NT* username and password with the appropriate permissions (usually *Administrator)*:

```
smbpasswd -j DOMAINNAME -r PDCNAME -U Administrator
```

It should now be possible to get lists of *Windows* users and groups with the *wbinfo* command:

```
wbinfo -u
wbinfo -g
```

To make the Winbind data available to the system, it is necessary to edit PAM and NSS configuration files. This should be done with great care, as it is possible to find oneself locked out of the system if these files are damaged. In **/etc/nsswitch.conf** add the word **winbind** to the **passwd** and **group** lines. In **/etc/pam.d/system-auth** add a line of the form:

```
auth sufficient /lib/security/pam_winbind.so use_first_pass
```

just after the equivalent **auth** line that uses **pam_unix**, and one of the form:

```
password sufficient /lib/security/pam_winbind.so use_first_pass
```

just after the equivalent **password** line that uses **pam_unix**.

It will be necessary to restart the Name Service Cache Daemon at this stage:

```
/etc/init.d/nscd restart
```

The translation of *Windows* usernames and groups into Unix-style passwd file format can now be seen with:

```
getent passwd
getent group
```

To automate the creation of user home directories on first login, add this line to the **session** part of **/etc/pam.d/system-auth**:

```
session required /lib/security/pam_mkhomedir.so skel=/etc/skel/
        umask=0022
```

(Ensure that the above is entered as a single line rather than the two lines as which it is presented here.) Note that this will create a separate Unix home directory for the user on each workstation that they use. It may also be useful to put a script into the **/etc/skel** directory to cause each user to automatically mount their *Windows NT* files in a standard place at login time.

14.6.3. Run GNU/Linux desktops in Active Directory domains

In principle, GNU/Linux desktop machines can join AD (Active Directory) domains in much the same way that they join *Windows NT* domains. Indeed, if the AD domain is running in NT-compatibility mode then exactly the same process can be used.

AD domains also offer the possibility of using LDAP for authentication and data lookup. This is the same scheme that is proposed for larger networks of pure OSS systems, and is well worth considering. By extending the AD schema to include Unix data, it would be possible to manage the users of OSS desktops and servers with AD administration tools. Storing the data centrally is preferable to the *Winbind* scheme used with *Windows NT*, as it keeps the mapping between *Windows NT* IDs and Unix IDs consistent across all machines.

14.6.4. Replace Windows NT PDC/BDC with Samba+LDAP

Samba can take on the role of the Primary Domain Controller, thus allowing all *Windows* servers to be eliminated even if some *Windows* clients are still required. Note that it is *not* possible to replace just the PDC or just a BDC in a domain: all domain controllers must be running the same system – either *Windows* or *Samba*. This is partly because the PDC-BDC replication protocol has not been reverse-engineered. Also, *Samba* Domain Controllers take a different approach to resilience: they delegate it to the LDAP servers where the data is actually stored.

Setting up a *Samba*+LDAP Domain Controller is too large a job to describe in detail here, but it can be done in a day or so by an experienced person. The larger task is planning the migration of usernames and groupnames from an existing domain. Some of the work is covered in the Samba-LDAP-HOWTO from IDEALX (see references in Section 14.12 below). The same source provides a set of migration tool skeletons that can be a very good base to build on.

In summary, the process is:

1. Install OSS server(s) with *Samba* and *OpenLDAP*. It may be necessary to build *Samba* from source, for instance *Red Hat Linux* 7.3 did not include the LDAP-enabled version.
2. Add the *Samba* schema definitions to the LDAP server.
3. Set up the LDAP server with an appropriate base Distinguished Name (DN) and directory tree structure (possibly using the tools from IDEALX to populate the tree with boilerplate entries).
4. Start *Samba* and test the Domain Controller function.
5. Use **pwdump** on the PDC to list all user entries in the SAM. Transfer the result as a text file to the OSS server.
6. Configure the IDEALX **smbldap-migrate-accounts.pl** tool to match the environment being built. This is non-trivial as there are a lot of options to consider.
7. Run **smbldap-migrate-accounts.pl** on the data transferred from the PDC. This will create entries in LDAP for all domain users. It will also set their SMB passwords to match the ones used under *Windows NT* (but this will not enable Unix or GNU/Linux logins, as the *Windows NT* passwords are hashed and a different hash scheme is used for OSS systems).

   The tool can create home directories at the same time if desired.
8. Copy user files and roving profiles from *Windows* servers to the new OSS servers, or re-bind the existing *Windows* servers to the domain now served by *Samba* Domain Controllers.

Large networks are likely to need multiple LDAP servers with data replication for resilience. If one *Samba* Domain Controller is associated with each LDAP server, a scheme very much like the *Windows* PDC/BDC setup can be realised.

There are many other issues to be considered, such as:

1. Choice of tools for user management
2. How *Windows NT* groups and ACLs will be mapped to Unix-style groups and ACLs
3. Whether to use a new domain name for the OSS-based service
4. How to create password hashes usable by OSS systems (or whether to continue using

Windows NT or *LANMAN* hashes, even in a pure OSS environment)

14.6.5. Replace Windows 2000 Active Directory with LDAP

The bulk of the data held by an Active Directory is in an LDAP-accessible store. At first sight, this should make it easy to replace AD servers with OSS equivalents. Unfortunately this is not the case: *Windows 2000* systems do not use pure LDAP for all data access, and they use a non-standard variant of Kerberos for authentication.

Several OSS teams are working to fix the problem, but at the time of writing the only feasible way to support *Windows 2000* and *Windows XP* clients is to run them in *Windows NT* domains as described above.

14.6.6. Run parallel GNU/Linux infrastructure and migrate users in groups

14.6.6.1. Replacing all Windows clients with GNU/Linux

This is the simplest of all possible migration schemes. The interaction between *Windows* and OSS systems is limited to the one-off transfer of user files. In outline, the process is:

1. Build the core OSS environment. This will include LDAP servers to hold configuration and username data, master installation servers, one or more file and print servers, and enough client workstations for systems management staff.
2. Build a development and training facility, with enough desktop workstations to allow training of appropriate-size groups of people. The initial job of this facility is to validate and fine-tune the workstation setup before the main rollout.

   At this stage, the workstation build process should be finalised so that machines can be set up with minimal human effort. It is very important that all desktop machines are installed in *exactly* the same way during the main rollout phase so this should be tested carefully.
3. Use the development and training facility in consultation with key representatives of the user base to generate enthusiasm for the project and to gather feedback on the user interface. Make changes as needed to arrive at the “rollout image”.

   Agree the training requirements and schedule.
4. Build a set of new desktop workstations sufficient to replace the equipment currently being used by the first group that is to migrate to OSS systems.
5. Register the first group of users on the new system.
6. Train the first group of users on the new system.
7. If necessary, re-set any configurations changed during training so that everyone starts with a known environment.
8. Replace the first group's desktop PCs with the pre-built OSS systems.

   At the same time, copy the group's files across to the new fileservers and set the original copy to be read-only.
9. Provide active support to the first group while they get used to working with the OSS system.
10. Upgrade the PCs removed from the first group as necessary, and install the standard

workstation image.

11. Repeat from step 5 with the next group of users.
12. When all users have migrated to OSS systems, make archive copies of all files on the old servers and decommission them.

14.6.6.2. Keeping some Windows clients

Where some *Windows* clients have to be retained (for example to support functions that are uneconomic to migrate due to non-portable software) there are two main options:

1. Retain a small *Windows* domain using one or more *Windows* servers.
2. Support the *Windows* clients from OSS-based servers using *Samba*.

The route chosen will depend on why the *Windows* clients are being retained, and on their geographic distribution.

In either case it is likely that *Samba* will be needed on one or more of the new servers, to provide file sharing between the *Windows* clients and the OSS-based ones.

14.7. Migrating server-side applications

14.7.1. Web servers: moving from IIS to Apache

The usual *Windows* web server is *IIS* (*Internet Information Server*), which provides HTTP, FTP, and Gopher services in one package. *IIS* has a reputation for problems with security and stability, which has prompted many organisations to replace it with an alternative. Indeed, after a series of particularly serious flaws was exploited in 2001, analysts at Gartner issued a strongly worded advisory to their customers suggesting that *IIS* should not be used for critical functions until it had been completely re-written by Microsoft.

There are a number of web servers to choose from when looking to replace *IIS*. Many of these are OSS or have very liberal licence conditions. Some of the more widely used servers are discussed in Section 11.4.2 above.

When migrating sites from *IIS*, the usual choice is *Apache* – often with PHP or Perl modules for scripting. *Apache* runs on GNU/Linux, *FreeBSD*, almost all other Unix variants, and also on *Windows*. This provides a wide choice of migration options.

14.7.1.1. Migration issues

1. Filenames and URLs

When moving a simple website from *IIS* on *Windows* to *Apache* on GNU/Linux or Unix, the main issue to be aware of is that the *Windows* filesystem ignores letter case in filenames, but most GNU/Linux or Unix filesystems are case-sensitive. As the hierarchy of web pages is normally represented directly in the filesystem, this means that URLs become case-sensitive when moved to the Unix or GNU/Linux environment. (This would not be an issue if *Apache* were used on a *Windows* server).

A less common issue is that *IIS* seems to accept both "\" and "/" as component separators – it must translate "/" to "\" for the *Windows* filesystem, but it appears to allow "\" to work natively. Thus, a file could be referred to in a URL as both **mydir\thisfile.html** and **mydir/thisfile.html**.

Neither issue will affect a correctly-written and self-consistent website. Unfortunately, sites built with *Windows* software often have inconsistent use of upper and lower case, and sometimes have the "\" character in URLs where the website file structure contains a subdirectory. Indeed, the example website distributed with early versions of *IIS* displays both of these issues. There are easy workarounds for both problems in *Apache*, which are demonstrated in the example later in this chapter. As a general rule though, it is better to correct such problems in the website data.

2. **Server-side image maps**

   Some early websites used a server-side mapping from x,y co-ordinates in an image to destination URLs. This is now deprecated because it is inefficient and does not work well with non-GUI browsers, but some sites may still use it. Server-side maps in *IIS* take the form of files with the "**.map**" extension, and their format is not compatible with the equivalent *Apache* files.

   The best approach is to convert any server-side maps into client-side maps as this also provides a better browsing experience for the user. If this is not possible, a simple Perl script can be used to edit the files into a form usable by *Apache*.

3. **Scripts and database connections**

   More complex sites are likely to have dynamic pages based on scripting and database access. Most *IIS* sites use *ASP* (Active Server Pages) as the scripting framework, and might use Access or *SQL Server* for the database, depending on the size of the application.

   There are many ways to handle the migration of *ASP* scripts. Some of the more popular ones are:

   1. *Chili!Soft ASP* package for Unix (now called *Sun ONE Active Server Pages*)
   2. *ASP2PHP*
   3. *Apache*'s *Apache::ASP* module
   4. Manual conversion to a new language

   Chili!Soft ASP is a proprietary product, but in some cases it could provide a very cost-effective migration route.

   ASP2PHP is a standalone script converter which converts text files written in *ASP* and *VBScript* into text files written in PHP. Support for *ASP* files using *JScript* is under development. PHP is a very popular web-scripting framework with similarities to *ASP*, so developers should find it a fairly easy transition to make. For larger projects it is often better to have a greater separation between page design and script logic than the *ASP* or PHP models allow. In these cases, a manual conversion using a templating system might be a better choice.

   Apache::ASP provides *ASP*-like features directly through the *Apache* framework, along with scripting in Perl. *VBScript* and *JScript* are not supported.

   In some cases it may be best to consider a manual conversion from *ASP* to a new framework. This allows the greatest flexibility, and complex sites may well benefit from moving to a template system such as *Template Toolkit* (http://www.tt2.org/).

   All the *Apache* scripting systems have database access facilities for a wide range of

database types (SQL, flat file, indexed, LDAP, NIS etc) so data-driven dynamic sites of any complexity can be built.

4. **FrontPage extensions**

   The *FrontPage* web-design package introduced a set of extensions to allow remote management of web content. These have since been used by some other web-design packages.

   The *FrontPage* extensions are available for Unix systems, but are not universally popular with *Apache* administrators for a variety of reasons including security issues and the large number of changes they introduce to the standard web-page storage area.

   A standards-based replacement is now available in the form of the WebDAV protocol (RFC2518). It is supported by most web servers (including *Apache*, using the **mod_dav** module) and is now the preferred website management protocol. Microsoft has supported WebDAV in their *Office* suite since *Office 2000*, and it can also be accessed directly using *Windows* Explorer, so a *Linux*/Unix/*Apache* server can support both OSS and proprietary clients using the same mechanism.

14.7.1.2. Migrating a static website

This example shows the complete process for migrating a simple static website from *IIS* on *Windows NT* to *Apache* on GNU/Linux.

1. Prepare the GNU/Linux server, connect it to the network, and test *Apache*. Most GNU/Linux distributions provide pre-configured *Apache* packages, so this is normally straightforward. An Internet-visible server will undoubtedly need its security tightened before being connected.
2. Locate the website data on the *IIS* server (usually in **C:\InetPub**) and make a copy of it ready for transfer, for example using a Zip archive package.
3. Copy the Zip file to the GNU/Linux machine (for example using FTP) and unpack it in the location chosen for the website data. This is configured as **DocumentRoot** in Apache's **httpd.conf** file, and is usually somewhere like **/var/www/html**.
4. Edit **httpd.conf** and add **default.htm** to the **DirectoryIndex** clause. (By convention, *Apache* is configured to look for default/home pages named **index.html**, while *IIS* uses **default.htm** – this action allows either name to be used.)
5. At this stage, the site should start to work, though it must be accessed by the name of the new server rather than the proper URL. It may also show problems where the site data has inconsistent use of upper and lower case in filenames and URLs, and where "\" has been used in URLs.
6. If possible, test the site at this stage and correct any problems by editing the site data. This will give the best performance. There are automated checkup tools available that will traverse the site and tell you if any links point to unavailable locations. You could also make a list of unreachable pages at this stage, and run every page through an HTML checker.
7. If fixing the site data is not feasible, add these configuration lines to **httpd.conf**:

```
LoadModule speling_module modules/mod_speling.so
AddModule mod_speling.c
CheckSpelling on
```

Note that this causes a directory scan and an HTTP redirect for each misspelled/mis-capitalised part of a URL, so watch out for performance issues.

8. Pages incorrectly using "\" in URLs can be handled using **mod_rewrite**, by adding these lines to **httpd.conf**:

```
RewriteEngine on
RewriteRule   ^(.*)\\(.*)$  $1/$2 [N]
```

This replaces the first \ with / in the URL and then repeats in case there was more than one \.

9. Check for server-side image maps using a command of the form:

```
find /var/www/html –name '*.map' –print
```

Edit by hand if there are just one or two, or use a script to fix them if there are many to do.

10. At this stage, the whole site should work correctly. You may want to set up FTP, *Samba*, or WebDAV to provide access for updating pages.

11. To bring the site into production, either disconnect the old server and change the IP address of the new machine to replace it, or change the DNS entry of the website to point to the new server.

14.7.1.3. A simple WebDAV configuration

WebDAV can be used to manage the content of some or all of your website. In this example it is used for the whole site, so no other access should be permitted. (Other management systems such as FTP or direct file access will confuse WebDAV clients as they do not use the same locking scheme.)

1. Make a directory for WebDAV locks. It should be owned by the same user and group that *Apache* runs as (see the **User** and **Group** configuration options in **httpd.conf)**. A good choice would be **/var/httpd/webdavlocks**.

2. Add these lines to the main part of **httpd.conf**:

```
Loadmodule dav_module libexec/libdav.so
Addmodule mod_dav.c
DAVLockDB /var/httpd/webdavlocks
```

3. Find the **Directory** or **Location** section associated with the default website, and add lines like this:

```
DAV On
AllowOverride None
Options Indexes
AuthType Basic
AuthName "Website Managers Only"
AuthUserFile /var/httpd/htpasswd
<LimitExcept GET HEAD OPTIONS>
  require valid-user
</LimitExcept>
```

4. Make sure that the associated files and directories are owned by the same user and group that *Apache* runs as, using a command of the form:

```
chown –R apache:apache /var/www/html
```

5. Create the password file:

```
touch /var/httpd/htpasswd
chown root:apache /var/httpd/htpasswd
chmod 640 /var/httpd/htpasswd
```

6. Create a password for a user called *webadmin* (or any other name you choose):

```
htpasswd –m /var/httpd/htpasswd webadmin
```

7. Either restart *Apache* or have it re-read its configuration files, for example:

```
/etc/init.d/httpd reload
```

8. You can now manage the whole site using the WebDAV protocol. *Windows 2000* and later clients can access it as a "Network Place" in *Windows Explorer*, and *Office* applications can save data directly to the site. GNU/Linux provides similar functions via **davfs**.

9. Note that the scheme described here provides only limited security. You should read the *Apache* manual for more details on user authentication and choose an appropriate scheme for your needs. It may be necessary to use SSL to secure the transactions; this can be done with *Apache*'s **mod_ssl**.

14.7.2. Databases: moving from Access and SQL Server to MySQL or PostgreSQL

Many small database projects on *Windows* use *Access*. This is an attractive product for many people because it is fairly simple to get started, and it has a familiar user interface. *Access* has severe limitations though; it was not designed for heavy multi-user work, and it cannot cope with large datasets.

Larger databases might use *SQL Server*, or one of the well-known relational databases: *Oracle*, *Sybase*, *DB2* etc. In the case of these larger systems, it may be best to leave the database running on the existing platform and just migrate the client applications to OSS platforms. This is particularly appropriate where the Administration has in-depth skills for the existing database and is using many proprietary features. There are standard ways to connect to relational databases across the network, so the choice of platform can be different for the database and the client applications. Also, most of the non-Microsoft proprietary databases are available on GNU/Linux and Unix platforms, so it is possible to change the operating system without having to learn a completely new database.

On the other hand, proprietary databases can be very expensive items so it is worth considering whether an OSS product could do the job effectively.

The two best-known OSS databases are *MySQL* and *PostgreSQL*. Both are mature products with large installed bases and active development teams. Both have good support for standard SQL, and are capable of very good performance.

It is also worth remembering that databases do not have to be relational. Some tasks fit better with other models, and direct use of an OSS product like Sleepycat's *Berkeley DB* can be extremely efficient. Similarly, the LDAP model of hierarchical networked databases is very suitable for some types of distributed application.

14.7.2.1. Migrating Access Databases

Access is only available on *Windows* platforms, so all such databases must be migrated to some other package if a completely OSS environment is planned. An interesting and useful intermediate scenario involves migrating the data to an OSS database but

continuing to use *Access* as the front-end. This has the desirable property of removing many of the restrictions and problems of the *Access* data-store.

1. **Hand-driven export/import**

   There are several ways to migrate data from *Access* to other databases. For simple datasets, perhaps the easiest way is to export the tables from *Access* as CSV (Comma Separated Values) files and then import these to the new server. This method does require the tables to be created by hand on the new server first, but it does not need any special software.

   As an example, here are the commands to create a database with a simple table and import a CSV file into *MySQL*. First enter at a shell prompt:

   ```
   mysql --user=myusername -p
   ```

   Then input the following:

   ```
   create database mydb;
   use mydb;
   create table mytable (
       firstname    char(30),
       surname      char(30),
       postcode     char(10)
   );
   load data local infile 'exportfile.csv'
       into table mytable
       fields terminated by ',' enclosed by '"'
       lines terminated by '\r\n';
   ```

2. **Scripted export/import**

   Several scripts and programs exist that will export an *Access* database complete with all information necessary to re-create the tables in another DBM. Some of these produce files to be copied to the new platform, while others connect directly across the network and make the changes immediately. An example of the file-writer scripts is **exportsql2.txt** available from http://www.cynergi.net/exportsql. This produces files with DROP TABLE, CREATE TABLE, and INSERT statements that will replicate the *Access* database in *MySQL*.

   A number of other migration tools are described in Paul DuBois' paper *Migrating from Microsoft Access to MySQL* (http://www.kitebird.com/articles/access-migrate.html).

   Once the data has been migrated, it is possible to continue using *Access* as a front end by deleting the tables locally and linking to the newly-created tables on the *MySQL* server.

14.7.2.2. Migrating SQL Server databases

The process here is much the same as described above; for simple databases it is usually sufficient to export the data to a common format (usually CSV) and then import it to the new database. More complex databases including stored procedures and triggers will need more effort, and in these cases it is well worth looking at the range of tools available to help the migration process. Some of these are OSS, and some are commercial. A few examples:

1. *PGAdmin* is free software for administering *PostgreSQL* databases. There are plugin utilities for it that handle migration of data from other database engines. More

information is available from http://www.pgadmin.org/.

2. *SQLPorter* from Realsoftstudio – a commercial product available in several variants depending on the source and target database engine. For more information, see http://www.realsoftstudio.com/overview.php.
3. *SQLWays* from Ispirer – a commercial product supporting a range of database engines. See http://www.ispirer.com/products.
4. *SQLyog* is another commercial tool – it does management on *MySQL* and also handles the migration of data from other ODBC-compliant databases: for details, see http://www.webyog.com/sqlyog.
5. The *MySQL* website lists a vast range of other conversion tools: see the list available at http://www.mysql.com/portal/software/convertors/index.html.

14.7.2.3. Database migration issues

Migrating the data is sometimes the easiest part of the job, though if data is to be accessed across the network as straightforward SQL-style tables then there is not much more to do.

The problems are most likely to come from all the ancillary utilities and scripting languages that surround any practical database. SQL itself is standardised, though almost all database vendors extend it and encourage people to use their non-standard extensions. Also there are often several different ways to achieve a given result in SQL, and the choice of which is most efficient may vary from one database to another.

Many database applications are built with application generators or form-builders. These may not work with databases other than the one they were sold with.

Both *MySQL* and *PostgreSQL* have developed enormously in the past few years, so it is important to make sure you read *recent* reviews when considering which to use and whether to migrate.

14.7.3. Groupware: moving away from Exchange

Exchange provides email, calendar and addressbook services. It is normally used with the *Outlook* client on *Windows*, though some installations also use *Outlook Web Access* (*OWA*) to provide basic functions through a web interface.

All the functions of *Exchange* can be replaced by OSS packages, often very efficiently. The problems come when trying to provide them seamlessly to *Outlook* clients, as the communication mechanism between *Exchange* and *Outlook* is proprietary. *Outlook* is capable of accessing certain open standards-based services, though in some cases the user experience is different from that found when using the proprietary protocol. As a result, it is worth deciding at the outset whether to migrate to an OSS client package at the same time as the server migration is done, given that the user population will see some changes even if they stick with *Outlook*. The most obvious replacement client is Ximian's *Evolution*.

14.7.3.1. General issues

All *Exchange* users will have usernames and passwords stored in the system. Recent versions of *Exchange* use Active Directory for this, so the notes elsewhere in this document about migrating user registration data also apply to *Exchange*. In summary, OSS-based servers can access registration data via LDAP, so the new servers can either use the existing Active Directory or the data can be migrated to an OSS-based data store such

as *OpenLDAP*.

14.7.3.2. Mail issues

Users may have considerable amounts of stored mail, both personal and shared with other group members. There may be a procedural or legal requirement to keep a log of all mail sent and received, in which case the storage and access to this data must be considered. People with portable computers may download all their mail to the laptop, or choose to maintain a synchronised copy with the master being held on the central store.

When planning a migration to OSS-based mail services it is important to locate all stored data and make sure it will still be accessible after the transition.

Exchange can use *Windows* groups as distribution lists – these are the same groups that *Windows* itself uses for access control. This is not the usual way of maintaining distribution lists in an OSS environment, but it can be supported if desired.

If *Outlook* is being retained as a mail client, it will need to be reconfigured to use IMAP rather than "native" access to mailboxes.

Exchange has no export facility so data migration must be done through a client connection.

For more detail on OSS mail systems, refer to Section 11.2 and Appendix C.

14.7.3.3. Addressbook issues

Outlook users build up a personal addressbook automatically as they send and receive messages. They also have access to one or more shared addressbooks if using *Exchange* server. The contents of these addressbooks must be migrated to an OSS-readable form. Personal addressbooks can be exported in vCard form, which is understood by many mail clients and can be parsed by scripts for conversion to other formats if needed. Similarly, shared addressbooks can be exported and then loaded into an LDAP store.

The main problems are likely to come from the fact that *Outlook* and *Exchange* tend not to use standard RFC822 mail addresses internally, so addressbook data may not include usable addresses when exported. In this case, some post-processing will be needed using a script with access to the Active Directory store to translate the "internal form" addresses to standard RFC822 addresses. This translation is likely to be needed even if *Outlook* is being retained as a mail client, as it will not be able to use "internal form" addresses when sending mail over standards-based protocols like SMTP.

14.7.3.4. Calendar issues

Some Administrations make considerable use of *Outlook*'s calendar facilities to arrange meetings and manage room bookings. These facilities can be used without *Exchange*, but there are some limitations.

If concurrent migration to OSS clients is planned, then calendars should be exported in vCal form and moved to the new calendar-management platform.

14.8. Migrating desktop applications to OSS

14.8.1. Office

14.8.1.1. Document conversion

OpenOffice.org is capable of reading and writing Microsoft's formats remarkably well, so it is not necessary to convert documents during the migration process. If document conversion is desired, this can be automated with the Autopilot feature selected from the File menu of *OpenOffice.org*. This provides a way of converting documents *en masse*. The decision to convert depends upon the future use of the document. Chapter 5 talks in general terms about document formats and conversion. If documents are going to be repeatedly edited then the format should be that used by the majority of editors.

14.8.1.2. Template conversion

OpenOffice.org can directly use templates in *Word 97* format, but in practice it is better to convert them to native templates and store them in an appropriate shared template area. This gives an opportunity to test each template and correct any conversion errors. *OpenOffice.org* itself does most of the conversion work, and the process can be automated for large collections of templates using the Autopilot Document Conversion function found under the File menu.

Templates from other word processors are likely to need re-creation by hand.

14.8.1.3. Macro conversion

OpenOffice.org uses a BASIC-like macro language. It is structurally very similar to the languages used by *Word* and later versions of *WordPerfect*. However, the names of the objects that it works on are different so all macros will need some manual conversion effort.

Macros in documents are a severe security risk and are not necessary for most day-to-day tasks, so it is well worth looking to see if they can be dispensed with. Most formatting tasks are better handled using templates and styles, and simple data manipulation can be done using forms.

Versions of *OpenOffice.org* from 1.1 include a macro recorder, making it easier to create simple macros if these are found to be essential.

There are no automated ways of converting macros at the moment although some work is being done on this.

14.8.1.4. Word Processing

There are many word processing packages in use on *Windows* systems. Well-managed organisations are likely to have standardised on one package, or may be in transition from one to another. The most common packages are:

- *Microsoft Word*
- *Microsoft Works*
- *WordPerfect*
- *Lotus AmiPro* and *Lotus WordPro*
- *Lotus Notes*
- *IBM DisplayWrite*

The OSS target is *OpenOffice.org*.

Files in *Microsoft Works*, *IBM DisplayWrite* and the Lotus formats are not directly

readable by *OpenOffice.org* so they would need conversion. It is often possible to export the files from the respective application in some common acceptable format; conversion may otherwise require a third-party tool.

WordPerfect files are not yet directly readable, but there is a project under way to include this format in *OpenOffice.org*. A script-based conversion program is available that could be used for bulk format conversion.

The website http://www.raycomm.com/techwhirl/magazine/technical/openofficewriter.html contains an extremely useful comparison of the functions available in *Word* and *OpenOffice.org*. The user-interface is similar enough to that of *Word* for people to swap from one to the other with little difficulty (though it would still be wise to arrange training to introduce the new package effectively).

14.8.1.5. Publishing

Document production beyond the capability of word processors is usually done with Desktop Publishing (DTP) packages. Common ones include:

- *Framemaker*
- *Pagemaker*
- *QuarkXPress*

The OSS product *Scribus* from http://web2.altmuehlnet.de/fschmid aims to replace these packages and may be worth evaluating.

OpenOffice.org is much more capable than word processors commonly were at the time DTP packages were first produced. Advanced features like Master Documents make it possible to handle large projects like book production, and page layout features are now very flexible.

Alternative approaches include the use of post-processing packages where the text is marked up in a language similar to HTML, and then converted to its final printable layout by applying style sheets. These non-GUI systems can be very useful for producing rapidly-changing documents and for on-demand printing from database-sourced material.

14.8.1.6. Spreadsheets

Common *Windows*-based spreadsheets include:

- *Microsoft Excel*
- *Lotus 123* and derivatives

Excel is by far the most commonly used.

The OSS target is *OpenOffice.org* although *Gnumeric* could also be considered.

In most cases, a migration from *Excel* or *Lotus 123* to *OpenOffice.org* or to *Gnumeric* should pose few problems unless the spreadsheet contains controls or other mechanisms requiring macros. In this case, these controls and macros must be rewritten.

14.8.1.7. Presentation Graphics

In a *Windows* environment, presentations are normally created using *Microsoft PowerPoint* or *Corel Draw*. *PowerPoint* with its ***.ppt** file format is the more common.

The OSS target is *OpenOffice.org*. It can read *PowerPoint* slideshows and templates with very few errors, and can be set to write ***.ppt** files as well if desired. As mentioned in Section 14.8.1.2 above, it would be worth bulk-translating important templates to the native *OpenOffice.org* format.

Users should be able to swap between *PowerPoint* and *OpenOffice.org* quite easily as the concepts and screen layout are very similar.

14.8.1.8. Graphics and image manipulation

Graphics packages divide into three main categories:

- Presentation graphics, dealt with in Section 14.8.1.7 above.
- Vector (line) graphics, typified by low-end MCAD programs and packages like *Microsoft Visio*.
- Bitmap graphics, including paintbrush programs and photograph manipulation packages like *Adobe Photoshop*.

1. Vector graphics applications

OpenOffice.org includes a drawing facility.

Dia (http://www.lysator.liu.se/~alla/dia)is an OSS package that is similar to *Visio*. It is much used for generating documentation diagrams, and has filters that will read files from older versions of *Visio* (not the 2002 version). There are symbol libraries for a range of applications. *Kivio* does a similar job and is designed to integrate well with the KDE environment, but appears more focused on flowchart diagrams. *Sodipodi* (http://sodipodi.sourceforge.net/) works well with SVG (Scalable Vector Graphics).

Files originated in *Visio* and similar packages may be readable by OSS software, but this should be tested in each individual case before planning a migration.

2. Bitmap graphics

This category ranges from simple programs like *Paint* right through to advanced image manipulators like *Adobe Photoshop*. The OSS world has spawned at least as many graphics programs as the proprietary sector, and with equally variable facilities and quality. One package stands out above the others though, and that is *The Gimp* (http://www.gimp.org/).

The Gimp is able to read almost all known bitmap graphics file formats (including *Photoshop*'s own internal format) and can generate most of them too. It provides all the features of a good paintbrush program along with layers, channels, and other advanced tools familiar to users of *Photoshop*. *The Gimp* is widely used in generating and enhancing images for the web and for publication. The only major feature that it currently lacks is full process colour management, so it may not be suitable for very high quality pre-press work.

14.8.1.9. PDF generation

Generating a PDF file in OSS is much easier than under *Windows* where something like *Adobe Acrobat* needs to be purchased. There are a number of Postscript and PDF tools available in standard distributions to this. In addition *OpenOffice.org* provides a way of directly producing PDF output. The production of a PDF can be set up as a print service thereby providing a method for continuing *Windows* users.

14.8.2. Mail

There is an enormous range of user interfaces for email, for both proprietary and OSS environments. As a result, this document can only provide a sketchy overview of the migration process and issues. Mail is also discussed in Section 11.2 and Appendix C.

The main client-side issues are:

- Choice of new Mail User Agent, and therefore its user interface
- Migration of existing stored personal mail
- Migration of existing addressbook entries

Whatever MUA is chosen, it will be necessary to migrate stored mail and addressbook entries. If the old MUA was set up to store all mail folders on an IMAP server then very little work is needed and the new MUA can simply be configured to access these in place. Where local files have been used as folders, it will be necessary to track these down and convert them. By default, *Outlook* stores mail in files with the extension "**.pst**" in **C:\Documents and Settings**<*username*>**\Local Settings\Application Data\Microsoft\Outlook**

Useful migration tools include:

- http://outport.sourceforge.net/ – export *Outlook* data to *Evolution* etc. This is under active development and by mid-2003 still had important limitations including the inability to export mail attachments.
- *Outlook*'s own export tool, possibly writing CSV or Excel format. This also suffers from being unable to export attachments to those formats.
- http://sourceforge.net/projects/ol2mbox – OSS tool for converting *Outlook* ***.pst** files to formats usable by OSS mailers. Does support attachments.
- *Kmailcvt* – OSS tool for converting some proprietary formats for use with *Kmail*.

14.8.3. Calendars and Groupware

Calendars, along with contact management and mail, are often grouped together under the general heading of Personal Information Management (PIM). Some integrated packages such as Microsoft *Outlook* provide all three functions in one interface, but others such as *ACT!* concentrate on contact management and could be regarded as closer to being CRM (Customer Relationship Management) systems.

The OSS target is *Evolution*, which integrates functions in much the same way that *Outlook* does. *Mozilla* could also be considered; it includes a capable email client and there is now a calendar module available from http://www.mozilla.org/projects/calendar – this is based on the open iCalendar standard and users can publish and share calendars using the WebDAV protocol.

14.8.3.1. Calendars

Some of the best-developed OSS calendaring facilities are in Web-based groupware suites,

and these are well worth looking at as possible organisation-wide services.

The iCalendar (previously vCalendar) standards define an exchange format for calendar entries. Details can be found at http://www.imc.org/pdi and in RFC2445, RFC2446, and RFC2447. Most OSS calendar suites can handle data in this format, so it is the preferred migration route as well as the normal means for day-to-day calendar management.

Some of the migration tools mentioned in Section 14.8.2 above can also extract calendar information from *Outlook* data files into iCalendar format.

14.8.3.2. Contact management

Almost every email package that has ever existed has defined its own format for the storage of address-book data. Many are restricted to storing just email addresses, but more recent formats tend to include all sorts of contact information. This diversity of formats makes migration more difficult than it would otherwise be.

Fortunately, the major email applications in both the proprietary and OSS worlds have tended to implement the iCard (previously vCard) exchange formats in recent years. The specification of this format is open, and can be obtained from http://www.imc.org/pdi and also found in RFC2425 and RFC2426. If contact details are to be transferred from a proprietary application to an OSS application then this is the preferred format.

Another way to handle contact information is to consolidate it into an organisation-wide directory and then access it via LDAP. This should certainly be done with widely used data such as the internal phone-book and email list maintained by many organisations. It is not a complete replacement for personal address-books though: an address-book should be small and focused on the requirements of the person using it, while a directory should be comprehensive and (probably) too large to browse through effectively.

Some of the migration tools mentioned in Section 14.8.2 above can also extract contact information from *Outlook* data files into iCard format.

14.8.4. Web Browsing

Windows users are likely to be using some version of *Microsoft Internet Explorer* for web browsing. It is also possible that some are using *Netscape*, *Mozilla* or *Opera*. The OSS target is *Galeon*, although *Mozilla* could be considered because it will run under *Windows*.

Migration from one web browser to another is quite easy for users, as they all have similar features and user interfaces (apart from text mode browsers such as *Lynx*, for obvious reasons). Issues for individual users will centre around the conversion of bookmarks: most OSS browsers can import bookmarks from *IE* and *Netscape* if installed on the same platform, but if the OS is being migrated as well then it may be necessary to export bookmarks in the form of an HTML file first.

Any organisation that uses intranet web pages should check that the HTML conforms to W3C standards so that it displays properly on all browsers. There are tools at http://www.w3c.org/ to help with this.

Any pages that depend on JavaScript will need particularly careful testing, as the dialect varies from one browser to another and the use of non-standard extensions will cause problems.

Any pages that depend on ActiveX controls will need to be re-designed to work in some other

way, as OSS browsers do not support this proprietary technology. ActiveX has a very poor security model, so disabling it is a valuable step in any case.

Common web plugin formats like Java, PDF, Flash, and RealPlayer are well supported by OSS browsers (though often using non-OSS plugin code which can sometimes be hard to find on the supplier's website). Other formats such as Shockwave Director will need to use CodeWeavers' *CrossOver Plugin*.

14.8.5. Personal Databases

People with data too large or complex for a spreadsheet but not large enough to justify a full commercial database often use *Microsoft Access*. This package provides a simple relational data store along with scripting and forms-building tools.

The migration of data to OSS databases is covered in Section 14.7.2.

There are advantages to storing databases on well-managed servers even if the central IT function is not managing the data or supporting the applications. An OSS migration provides an opportunity to offer such a data-storage service, by setting up a server where individuals can build their own database applications. There are several web-based packages that could be chosen as a basis for this, such as *PHPmyAdmin*: http://www.phpmyadmin.net/documentation has details.

Tools with more conventional GUIs include:

- *Kexi* (http://www.koffice.org/kexi) – a database front-end from the KDE project, aimed at a similar market to *Access*.
- *DBDesigner* (http://www.fabforce.net/dbdesigner4) – a tool for more advanced users, integrating with both Gnome and KDE.
- *Knoda* (http://www.knoda.org/) – another simple front-end for KDE.

None of these tools attempts to read *Access* files.

14.9. Migrating print services to OSS

In small office environments it is common for printers to be attached directly to desktop workstations. Larger offices, and those with a requirement for high-volume printing, are more likely to use networked printers – these may be directly attached to the network or driven by a print-server.

OSS environments support both models, though it is more common to find print-servers and a small number of high-capacity printers.

14.9.1. The Windows print model

Printing in *Windows* is almost always done from a menu item in a GUI application. It is very rare for the command-line to be used. Applications generate printed output using a process very similar to the one they use to put information on the screen. A set of printer-specific drivers are then used by the operating system to generate the actual data stream for the printer. These drivers are normally supplied by the printer manufacturer, and must be installed locally or on the print-server before any printing is possible. In a networked environment it is normally best to install and configure the drivers on the print-server so that clients do not need to be hand-configured. (It is still necessary to connect the printers to the client: this can either be done by hand or by a login script)

14.9.2. The Unix and GNU/Linux print model

GNU/Linux inherited its print model from BSD Unix. Applications generate files or streams of print data which are passed to a print spooler which takes responsibility for the print job. Jobs may be queued and can be passed transparently to other machines on the network. Early Unix systems had no printer-independent interface for generating print data, so each application had to include code for every sort of printer it wanted to drive. In the days of character-only printing this was not a problem, but when manufacturers started adding graphics facilities they each created a new and different printer language.

BSD print systems have always had the ability to feed print jobs through a set of filters, so people started to write filters that would convert from one printer language to another to increase the range of supported printers. Many of the better printers used in research labs had PostScript interpreters, so PostScript came to be used as the common printer-independent language.

Most GNU/Linux distributors are now replacing the BSD print system with a new package called *CUPS* (Common Unix Printing System), that supports the Internet Printing Protocol (IPP) in addition to the traditional **lpr** protocol. This completes the transition to the new print model:

- Applications generate print jobs in PostScript.
- When the jobs are passed to the print system, the application can request any special features that the printer supports (duplex printing, folding, stitching, punching, binding etc). The requests have a standard format, but obviously they will only succeed if the printer has the necessary hardware. There is a standard way for applications to find out what features are supported by a given printer.
- Jobs may be spooled locally on the workstation or passed immediately to a print server. The user does not need to be aware of which method is used.
- The print system may spread jobs across a number of similar printers in a print farm.
- The print server runs the job through a pipeline of filters to convert it in stages to whatever format is needed for the actual printer, and to control the communication with the printer.

There are now over 600 printer models known to work perfectly with this model (i.e. GNU/Linux applications can access all functions that are available using the manufacturer-supplied *Windows* drivers, and with equivalent or better results).

Although PostScript is the most commonly used intermediate format, *CUPS* can be configured to support almost any file format that filters are available for. In particular, it is common to allow PDF, JPEG and some other formats to be printed directly, and some sites add filters to do automatic pretty-printing of email etc.

CUPS provides interfaces compatible with BSD's **lpr** suite and also with System V's **lp**. It is therefore possible to replace the older systems entirely on existing machines (*FreeBSD*, *OpenBSD*, and most commercial Unix variants). A port for *Windows* is under way.

CUPS provides a vast range of features and facilities, including auto-discovery of print servers, page accounting, quotas, etc. For full details see the *CUPS* website referenced below.

14.9.3. Setting up an OSS-based printing service

For very small deployments it is simple to set up directly-attached printers on each client

workstation. These can be shared across the network if desired, and *CUPS* supports this very easily.

The use of print-servers is recommended for all cases where there are more than a handful of clients, or where there is a substantial print volume. One or more print-server machines should be installed, and given logical names in the DNS in addition to their natural hostnames. This allows configurations to reference names like **printserver.example.org** rather than **pc35.example.org**, thus making it much easier to re-organise the service later on. All client machines should be configured to refer to one of the printservers for all printing requirements: this avoids having to re-configure any clients when printers are added or removed.

14.9.4. Printing from Windows clients to GNU/Linux-attached printers

There are several ways to set up GNU/Linux-based print servers to support *Windows* desktop machines. These vary in the amount of initial effort and the amount of per-client effort required.

14.9.4.1. Using the lpr protocol

This method is appropriate where a very small number of *Windows* clients needs to be supported.

lpr is a very common protocol for passing print jobs between Unix machines. As mentioned above, it is gradually being replaced by IPP, but it is widely implemented and can be used from most versions of *Windows*.

1. Make sure the GNU/Linux machine is configured to accept jobs using **lpr**.
2. Obtain a suitable set of *Windows* drivers for the printer. Ideally, this should be the generic *CUPS* driver which generates portable PostScript, but it is possible to use printer-specific drivers if *CUPS* is set to permit raw printing.
3. Log onto the *Windows* machine as Administrator.
4. Open the Networking utility in the Control Panel, select the Services tab, and make sure that “Microsoft TCP/IP printing” is listed. Add it if necessary (this will require the distribution CD and a reboot).
5. Configure the printer on the *Windows* client as if it were a local printer (not networked!). When selecting a port for the printer, create a new “LPR port” and configure it to send the jobs to the GNU/Linux server.

The *Windows* client will now be able to send print jobs to the GNU/Linux machine, but *Windows* tools may not be able to see and manipulate queued jobs. *CUPS* supports web-based management, so users should be advised to use that where necessary.

14.9.4.2. Using printer shares

This method is also appropriate for small numbers of *Windows* clients. It works with *Windows 95/98/ME* as well as *Windows NT/2000*.

1. Install and configure *Samba* on the GNU/Linux server. Follow the instructions for creating printer shares: it is easy to have *Samba* automatically create a share for each printer that the server knows about.

2. On each *Windows* client, use the Add Printer Wizard to add a networked printer. You should be able to browse a list of servers to find the one you want to use. You will need to install printer drivers locally on the client machine.

As a refinement of this scheme, it is possible for an administrator to use the Add Printer Wizard to upload *Windows* printer drivers to the *Samba* server, so that they do not need to be installed individually on the clients.

14.9.4.3. Using Point and Print configuration

This method is appropriate for larger installations and those where new client machines must be configured by less-skilled staff. It requires rather more effort to set up in the first place, but is much easier to use once done. The process is quite involved, so please refer to the *Samba* HOWTO collection for full details.

1. Install and configure *Samba* on the GNU/Linux server. For full Point and Print support this must be version 3.0 or above, though many functions are supported in 2.2.4. Make sure *Samba* is built with *CUPS* support.
2. Configure *CUPS* to support *Windows* printers by adding the *CUPS* driver package.
3. Use **cupsaddsmb** to install the *Windows* drivers from *CUPS* into *Samba*.
4. Connect from a *Windows* client using an ID that has permission to modify print settings on the server, and set the default printer characteristics appropriately (page size etc). This is more tricky than it sounds, as *Windows* provides two identical windows in different parts of the configuration GUI and only one of them affects the default settings. (See the *Samba* HOWTO for details).
5. On each *Windows* client, browse the network neighbourhood to find the server. Right-click on the printer you want and "Connect" it. The printer now appears in the local printers collection and can be used easily.

Large installations will want to use a logon script to perform step 5 rather than doing it by hand.

14.9.5. Printing migration schemes

For small sites with a handful of workstations and printers, it is simple to set up a GNU/Linux-based print server and just re-configure each client workstation by hand. If there are several shared printers attached to desktop machines it may be worth taking the opportunity to consolidate them onto a print server. This can be made easier by fitting ethernet cards to printers where they support the facility (this can also provide a substantial performance improvement over serial or parallel interfaces). Parallel-only printers can be networked using network-printer boxes.

Larger sites will certainly benefit from the use of one or more print servers. These machines can perform other tasks as well, but if there is a substantial print volume it should be remembered that converting PostScript to other formats is a CPU-intensive job and machines should be sized accordingly. It will be worth setting up the full Point and Print configuration if *Windows* clients are to be supported, as the migration of client machines from the old *Windows* print servers to the new GNU/Linux ones can then be done with a simple login script.

14.9.6. Potential Problems

There are some common problems which can be avoided by careful planning:

1. Make sure each printer is controlled by one server only. Make all other desktops and servers send print jobs for a printer via its controlling server. This is particularly important with network attached printers. If this is not done then the printer could receive two or more print jobs at the same time with possible corruption of output.
2. If possible, try to arrange for only one set of drivers to format the output for a printer. This is probably best done on the controlling server, but not necessarily – it depends to some extent on which server has the best driver for the printer. Other machines spooling the output should treat it as raw data. If this is not done then sometimes a driver will try and format an already-formatted output stream, corrupting the output. This is only likely to be a problem when the formatted output contains binary data.

14.9.7. Further information on printing

A great deal of information is available on the Web. These sites are useful starting points:

http://www.cups.org/ – *CUPS*, the Common Unix Printing System.

http://www.linuxprinting.org/ – the Linux Printing site, with a vast range of useful information.

http://www.linuxprinting.org/kpfeifle/SambaPrintHOWTO – the *Samba* Printing HOWTO. Note that this is the author's distribution site and the document is a work in progress. Look for the latest draft.

14.10. Legacy applications

Those applications which have no OSS alternative and cannot be re-compiled to run under OSS will need to be run on a machine running the legacy operating system, or under a hardware or software emulator. The techniques discussed in Chapter are applicable.

14.11. Virus protection

An up-to-date anti-virus package is now essential in *Windows* and *Macintosh* environments. Even organisations with little interest in security ignore this protection at their peril.

By contrast, there are very few viable viruses that affect OSS systems. As a result, virus protection in OSS environments is usually limited to scanning email to avoid passing on viruses to *Windows* users.

There have been automated attacks on OSS systems in the past, of which the most famous was the 'Morris Worm'. A strong focus on security in the years since that event has reduced the risks considerably, but it is still possible that an effective virus could one day be released. Good systems management practices and continuing user education are currently a better defence than anti-virus software.

There are currently two known OSS anti-virus projects, *Open Anti Virus* (http://www.openantivirus.org/) and *ClamAV* (which has apparently vanished from the net). Both are in very early stages of development and neither is recommended for service yet. Many of the commercial anti-virus products have versions that will run on OSS platforms. These versions are not entirely equivalent to their *Windows* counterparts – at present they are aimed at functions like mail scanning rather than on-the-fly virus detection in running code as is common in the *Windows* environment. However, as explained above, on-the-fly detection is mostly unnecessary on OSS systems; mail scanning is generally sufficient.

14.12. References

http://www.samba.org/	The *Samba* SMB file/print/domain server
http://www.openldap.org/	The *OpenLDAP* directory server
http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam.html	The Linux-PAM system administrator's guide
http://samba.mirror.ac.uk/samba/docs/man/Samba-HOWTO-Collection.html	The main *Samba* HOWTO collection
http://www.csn.ul.ie/~airlied/pam_smb	pam_smb PAM module to authenticate GNU/Linux users with SMB
http://samba.idealx.org/index.en.html	IDEALX tools and HOWTOs relating to *Samba* (some in French)

15. Scenario 2 – Unix

The Administration has Unix servers ("big Unix" – Solaris, HP/UX, AIX, OSF/1, etc). Uses PCs with client-server applications. Some Unix Workstations and X Terminals.

Migrating the PC desktops will be similar to Scenario 1 above. The Workstations and X terminals are likely to be running X based applications which should run without any problem on the new OSS desktops. The main problem here is migrating the servers.

Migrating from Unix to GNU/Linux is similar to porting from one version of Unix to another. Bearing in mind that the term Unix includes the AT&T, BSD and the OSF/1 code bases which are different implementations of the POSIX standard – as is GNU/Linux. The differences lie when a program uses features that are outside POSIX, typically things like system administration and performance enhancing features. Poorly written programs that have been written with little regard for POSIX will also have problems – writing portable programs is an art, "home grown" programs are likely to need some work (elimination of all compiler warnings produced at the highest warning level is a good start). However the problems are likely to ones of detail rather than fundamental architectural ones. Unixes make use of open protocols such as TCP/IP and use common services such as DNS and DHCP.

Configuration is also likely to be different. However the system data is unlikely to be held in a proprietary format and so manipulating it to conform to GNU/Linux requirements should be fairly easy. This includes usernames and passwords although subtle differences may mean that simple transfer is not possible.

If source code is available then recompilation should allow the code to be ported. However there are some issues which will need to be addressed:

1. There is no standard for the location of files and these maybe hard coded into programs (such as **/usr/bin**, **/usr/local/bin** or **/opt/bin**).
2. There may be different values for system constants for instance the maximum number of files that can be open at any time.
3. Subtle differences in the programming language, for instance ksh and pdksh. Different C compilers are more or less strict on syntax checking, therefore code which would be allowed on one machine might give an error on another.

   The code could be unportable because of for instance:

   - Non portable use of constants, such as using number rather than SIGPIPE (something that is defined in a C header file). This is an example of a programmer programming to the operating system rather than the POSIX standard.
   - Assumptions about word length or byte ordering.

   `gcc`, the compiler on GNU/Linux, has options to be quite flexible in these circumstances.
4. Each Unix may have different header files and libraries. They may also be in different locations. The locations and names can be changed automatically once they have been found. However if the library or header file provides different behaviour then manual intervention is needed. For instance:

   - Semantics of some library calls differ:
     - such as threads,
     - exec (setuid bit on scripts ignored),

- asynchronous I/O,
- ioctl for tty control.

- Different values for errno.

The original code may use proprietary applications or libraries that are not available under GNU/Linux. It may have to be re-written to use what is available on GNU/Linux. This could be the case if special hardware interfaces are required, for instance a fax card. There is possibly much work to do in these circumstances.

5. The makefiles which help build applications may need to be updated to reflect the differences mentioned above.
6. Applications may make assumptions about specific subsystems such as printing and databases. This means for instance that SQL code may have to be re-written.
7. Porting any code to a new hardware, compiler or operating system can show bugs in the program that were always there but just never showed themselves, for instance because memory is laid out differently, integers have different sizes or bytes are ordered differently.

   The following references give more detail.

   http://www.linuxhq.com/guides/LPG/node136.html

   http://www1.ibm.com/servers/esdd/articles/porting_linux/index.html?t=gr,l=335,p=PortSolaris2Linux

   http://www-106.ibm.com/developerworks/linux/library/l-solar/?open&t=grl,l=921,p=Sol-Lx

   http://www-1.ibm.com/servers/eserver/zseries/library/techpapers/pdf/gm130115.pdf

   http://www.unixporting.com/porting-guides.html

16. Scenario 3 – Mainframe

The Administration is mainframe-based (it might be running MVS, VM/CMS, AS/400, or even Unix). Most users have green-screen access. There are a few PCs, mostly being used as terminal emulators but with one or two local applications.

This Scenario is similar to the thin client one as far as the desktop is concerned, particularly if the architecture is to remain.

There is no data on migration of the servers was available.

17. Scenario 4 – Thin client

The Administration uses thin desktops with access via *Citrix* or similar to a mixture of Windows and Unix applications.

The use of the BRA is not assumed here as the original reasons for using a thin client model are still likely to hold. However if a move to the BRA is contemplated then many of the same problems in Scenario 1 will arise. Migration in this Scenario under this assumption is therefore very simple as the architecture is not going to change.

Because the client is very thin all that is required is an OSS viewer for each protocol required. The windowing system doesn't need to have a great deal of functionality so a light weight manager like *tvwm* would be sufficient.

The following protocols can be supported (among others):

1. HTTP. Any OSS browser would be sufficient. The ability to run Javascript and Java would need to be investigated. In addition, any plugins required would have to be supported directly, through a substitute using the *plugger* package or through the proprietary CodeWeavers *CrossOver Plugin* package.
2. ICA. This is the proprietary *Citrix* protocol. *Citrix* provides a zero-priced, but non-OSS, ICA viewer which works under GNU/Linux.
3. RDP. This is the protocol used by *Windows Terminal Server*. An OSS viewer for RDP, *rdesktop*, is available.
4. VT220, VT100 etc. These DEC protocols are all supported by *xterm* using the appropriate TERM environment variable setting. Connection to the host is made via telnet.

   xterm can emulate many different terminal types by changing the value of the TERM variable. For instance setting **TERM=prism9** will emulate the protocol used by PRIME machines. All the emulations assume telnet connectivity or similar and a character-based rather than page-based protocol.
5. 3270. The *x3270* program provides the appropriate support. Connection to the host is made via telnet.
6. X. This is the native display protocol on GNU/Linux and so should provide no problems.

There are proprietary products for some of the more esoteric protocols.

The Linux Terminal Server Project (LTSP) http://www.ltsp.org/ provides a number of kits to build thin client devices based on GNU/Linux. This is an extremely active project and the quality of the software seems to be very good.

The changes required on the servers are similar to the considerations discussed under other Scenarios.

Appendices

Appendix A. Publicly Available Case Studies

A.1. http://www.turku.fi/tieto/liite44.rtf

City of Turku testing of:

1. OpenOffice.org,
2. x3270,
3. IBM Host On-Demand,
4. WRQ Reflection for the Web,
5. Hansa,
6. AS400 Emulator,
7. Netscape Communicator,
8. Mozilla,
9. Konqueror,
10. F-Secure Anti-Virus.

No actual migration experience reported. Intention to migrate in 2003.

A.2. http://www.m-tech.ab.ca/linux-biz

A collection of sites apparently using GNU/Linux. Information on each site is slight, but there are links to possible contacts who may be able to provide further information.

A.3. http://www.washingtonpost.com/ac2/wp-dyn/A59197-2002Nov2?language=printer

A description of the use of OSS in Extremadura in Spain. A local distribution has been created called linex (see http://www.linex.org/). Their experience is that users have little problem in using the software. Also users need access to `Windows` to deal with Microsoft's file formats in some circumstances. They have recently announced that 80,000 users are now supported.

A.4. http://newsforge.com/print.pl?sid=02/12/04/2346215

This is a story about the City of Largo. It has a number of links and discusses the cost of ownership.

A.5. http://people.trustcommerce.com/~adam/office.html

A company that has written up its experience of moving to OSS on the desktop as well as their servers. A KDE site.

A.6. http://www.business2.com/articles/mag/print/0,1643,44531,00.html

A description of Zumiez who have put Ximian Gnome desktops into their organisation. Other case studies as well.

A.7. http://lwn.net/Articles/13301/?format=printable

This contains reports of experiences in Denmark. It refers to a report by the Danish Board of Technology which unfortunately is not fully translated into English.

A.8. *http://www.siriusit.co.uk/support/casestudies/k_g_case.html*

A case study of a UK company which migrated to OSS recently.

A.9. *http://staff.harrisonburg.k12.va.us/~rlineweaver*

An experience of using OSS in schools.

A.10. *http://www.li.org/success*

A list of case studies. Some useful information.

A.11. *http://www.statskontoret.se/pressrum/press/2003/press030207english.htm* *http://www.statskontoret.se/pdf/200308eng.pdf* *http://www.statskontoret.se/pdf/200308engappendix.pdf*

A study of OSS in public administrations in Sweden.

A.12. *http://www-3.ibm.com/software/success/cssdb.nsf/topstoriesFM?OpenForm&Site=linuxatibm*

A list of IBM-specific case studies. No real detail and server-centric but gives a feel for the real world use of GNU/Linux.

A.13. *http://h30046.www3.hp.com/search.php?topiccode=linuxCASESTUDY*

A list of HP-specific case studies.

A.14. *http://openapp.biz/seminar/Tony_Kenny/Tony_Kenny.pdf*

Beaumont hospital in Dublin has over several years moved completely to OSS not only in its technical but also its administration IT. The link gives quite a lot of detail. Beaumont expect to save on the order of €13 million over a period of 5 years due to the change. They have found that, properly managed, the migration is found to be acceptable to all staff and that the OSS systems can be run much more efficiently.

Appendix B. Wine

Wine stands for "Wine Is Not an Emulator", and full details can be found at http://www.winehq.com/.

B.1. History

The development of *Wine* started around 1993 by Bob Amstadt who was running GNU/Linux and *Windows* on the same machine. GNU/Linux software had reached the point where it was capable of meeting most of his needs, but he had some games programs he was very fond of which were only available on *Windows*.

Fed up of re-booting just to run games he started work on a means of intercepting the system calls the games used and mapping them to the GNU/Linux X environment. Others heard of the work and chipped in to help, until *Wine* was capable of running the games they wanted to play.

Around 1995 people tried to run other programs, including *Quicken* and the *Office* suite, and these applications then became the main area of interest. A separate group split off, continuing the support of the complex graphical techniques used in games but which are in general not used in office style applications. The project now took on a more formalised approach, with a team of core developers and a project leader. Since 2000, the project has been put on an even more systematic footing, with a project leader and support team based in the USA, two small development teams in Canada, and developers in most European countries. Major vendors are also contributing, for instance IBM.

Techniques were created to identify the operating system calls that the programs made. In most cases the development of a small amount of code would enable a given application to run. It was found that programs often make preparations to call a particular interface but then do not actually make the call. Code was therefore written to allow the programs to continue making these preparatory calls without getting immediate errors, as well as code to support actual calls.

The first commercial use of *Wine* was by Corel, who had done a lot of work in supporting *Wine* and used it to produce a GNU/Linux native version of *Wordperfect 8*. Other companies have since used *Wine* to produce a GNU/Linux version of their products with the minimum of effort and change, one of the latest being Xilinx who produce specialist electronics CAD packages. The Ximian *Mono* project is likely to use *Wine* to allow .NET applications written for *Windows* to work without being rewritten. See http://appde.winehq.com/ for details on the level of support for a range of applications.

Recently a team of experienced *Windows* application developers has started to produce a suite of test programs to systematically check the 12,000+ system calls currently in the *Windows* library set.

Currently *Wine* comprises around 750,000 lines of "C" code implementing around 90% of the calls in popular *Windows* specifications such as ECMA-234 and Open32. Calls that are not publicly documented are more difficult to implement, but progress is being made.

Some companies working on *Wine* develop code for particular functions which initially is proprietary. They do this to fund themselves and their work on the mainstream project. They move their code into the mainstream when they have a suitable alternative source of revenue. Support for OLE and ActiveX falls into this category.

B.2. What Wine does

Wine intercepts all *Windows* and DOS system calls together with BIOS interrupts and tries to map them into the GNU/Linux X environment. Native processor instructions are executed as they would have been in the the *Windows* environment, and therefore *Wine* is not a full emulator.

Wine is not tied to the Intel x86 architecture – for instance, a version for the DEC/Compaq Alpha exists, but serious demand and usage only exists on x86. It does not enable x86 *Windows* programs to be run on another architecture such as the PowerPC or SPARC, although *Wine* will compile and run on both.

Not every interface in the *Windows* environment can be mapped onto an interface in the GNU/Linux and X environment. Some *Windows* interfaces simply do not have an equivalent. This means that in some instances a significant amount of code has to be written to support the mapping. For instance, there are problems with the more complex cursors used by some *Windows* programs. The X Window System cannot cope with more than 2 colours in a cursor, which means that *Wine* has to make assumptions about what colours to use, occasionally with unusable results.

Wine is actually two products, *Wine* itself, which allows pre-compiled *Windows* programs to run, and *Winelib*, which may be used to compile a *Windows* program to produce a native GNU/Linux program (this is what Corel used to produce the GNU/Linux version of *Wordperfect*).

Winelib could be used to run programs on hardware other than x86 if the source code is available, although other architectures specific problems may still exist (for instance endian issues).

B.3. What Wine is good at

Support is available for *Windows 3.x/95/98/ME/NT* programs (although *Windows NT* support is less complete). Many programs intended for *Windows 2000* will run, unless they use specialised new interfaces introduced with *Windows 2000*. Little work has yet been put into supporting specific *Windows XP* programs, but there are very few of them yet.

Wine supports most of the *Windows* publicly documented interfaces, however, support is not always as complete as one would like. See http://www.winehq.com/?page=status for details on the current state of support in *Wine*.

Programs that run in isolation or only use external communications interfaces will normally run. Each program should be individually checked because the precise interfaces and parameters used could interact to cause problems.

Some people have reported running compilers and development environments with some success.

B.4. What Wine is not good at

Some specific areas are not complete, for example Dynamic Data Exchange (DDE). However many programs make DDE calls without actually using them, and hence will work quite happily. OpenGL and other specialist high-speed graphics areas also have issues. Implementation of Access Control Lists (as in *Windows NT*) exists in part, but has not yet been integrated with ACLs in the underlying O/S.

VxD device driver technology, introduced with *Windows 98*, is a difficult area. This needs access to the hardware and kernel internals in a manner that any serious multi-user system

could not allow. There are techniques available to produce equivalents but they involve a lot of work and are not guaranteed to work. In some cases the vendor can be persuaded to produce a native GNU/Linux version that uses normal GNU/Linux interfaces. As this architecture is being dropped by Microsoft (*Windows NT* type architectures will not allow such access) this will slowly cease to be a problem.

Some *Windows* programs try to manipulate devices directly (especially serial ports). This is not allowed in GNU/Linux, or any other Unix variant. This normally only applies to communications packages such as *Procomm*, and programs derived from DOS products where such things had to be done.

Rendering of some graphic images is not yet satisfactory, especially TrueType font handling. However this is being actively pursued.

The other difficult area is software developed by Microsoft themselves. This is because these products tend to use undocumented interfaces. While it is possible to discover what is happening, developers have to be careful since the laws on reverse engineering are very strict in some countries. The USA, for example, forbids reverse engineering for any purpose, and most other western countries allow it only for the establishment of compatibility. Thus, work in this area will always be rather slow.

Running application installers, in particular, has been problematic, but recent work has resolved most of the difficulties, and work is continuing. Some of the difficulties are caused by package developers not using recommended techniques. Access to the registry is one example of this. *Wine* holds its registry in a different format to *Windows*, to make recovery easier. As long as the documented interfaces are used to access the registry this does not matter, but developers sometimes, at risk of corrupting a real *Windows* registry, access the registry directly, with the result that the program cannot be made to work under *Wine*.

Wine is sometimes criticised for poor performance, but this is often due to extensive debugging code. It is possible to compile *Wine* without this, but this should be done with care as it means that problems cannot be diagnosed without further recompilation.

B.5. Wine – commercial alternatives

As mentioned earlier, extended versions of *Wine* are available as commercial products to support development of the mainstream of *Wine*. Two companies doing this are Transgaming and CodeWeavers. Transgaming mainly work on improving graphics and sound interfaces and their product is aimed at the games market. CodeWeavers are working on mainstream office applications and have a product, *CrossOver Office*, which, for instance, supports *Office* and *Lotus Notes*.

B.6. Wine and Visual Basic

Visual Basic 3 has been reported to run under *Wine*, but no details are available.

Visual Basic 6 currently will not install. Work is under way to tackle this, but it is too early to tell whether they will be completely successful or not.

No other versions are known to have been tried.

B.7. Application Migration to Wine

This is a list of general guidelines to manage the process of migrating applications to GNU/Linux under *Wine*:

1. Check the licence conditions. Some companies have issued licences that forbid running their application except on the target operating system. For instance, Oracle used to do this and Microsoft are starting to do it for components that can be freely downloaded; the situation is fluid. Remove any program with such conditions from the test list and make a separate list of them.
2. Obtain copies of all applications that are to be migrated (this may well be all). Site licences may not allow copies for testing purposes.
3. Set up a test machine with the latest snapshot of *Wine*.
4. Test each and every program on the test list. Note all problems encountered, also note if they are in the install, initialisation or running phases. Also assess if they actually affect what users need to do by testing with a representative selection of end users. Poor performance should also be noted. Warning messages will be output indicating where system calls are not yet implemented or are incompletely implemented.
5. For each program in the problems list, first check if there is already a GNU/Linux implementation. If so, there should be no problems, but test as far as possible. If there is no existing GNU/Linux implementation it will be necessary to contact the vendor and suggest creating one by the use of *Winelib*. Again, DLLs may be missing. Each vendor will have to be dealt with separately.
6. Where vendors refuse to cooperate then alternative applications will have to be found, or the project abandoned.
7. Once a list of the extra DLLs and library calls required is available it will be possible to get a price for the implementation.
8. Each program will need retesting with new snapshots of *Wine/Winelib* until the problems all vanish. Patches sometimes cause problems with programs that were previously running correctly, and this needs to be tested.
9. *Wine* is normally compiled with debugging tracing, and this hits performance badly, especially in screen interactions. Any programs that run correctly but have performance problems should be re-run against a copy of *Wine* compiled without the debug tracing. If performance is still unsatisfactory then development work will be required.

Appendix C. Mail Systems

This appendix details mail systems in general because the range of OSS mail products can sometimes be confusing and the terminology used is not always clear.

The Internet Mail Model is based on a number of logical components, each of which has a specific job to do and communicates with the others through the use of open protocols. This model is the one used by OSS systems. The model can be best described with the help of some diagrams.

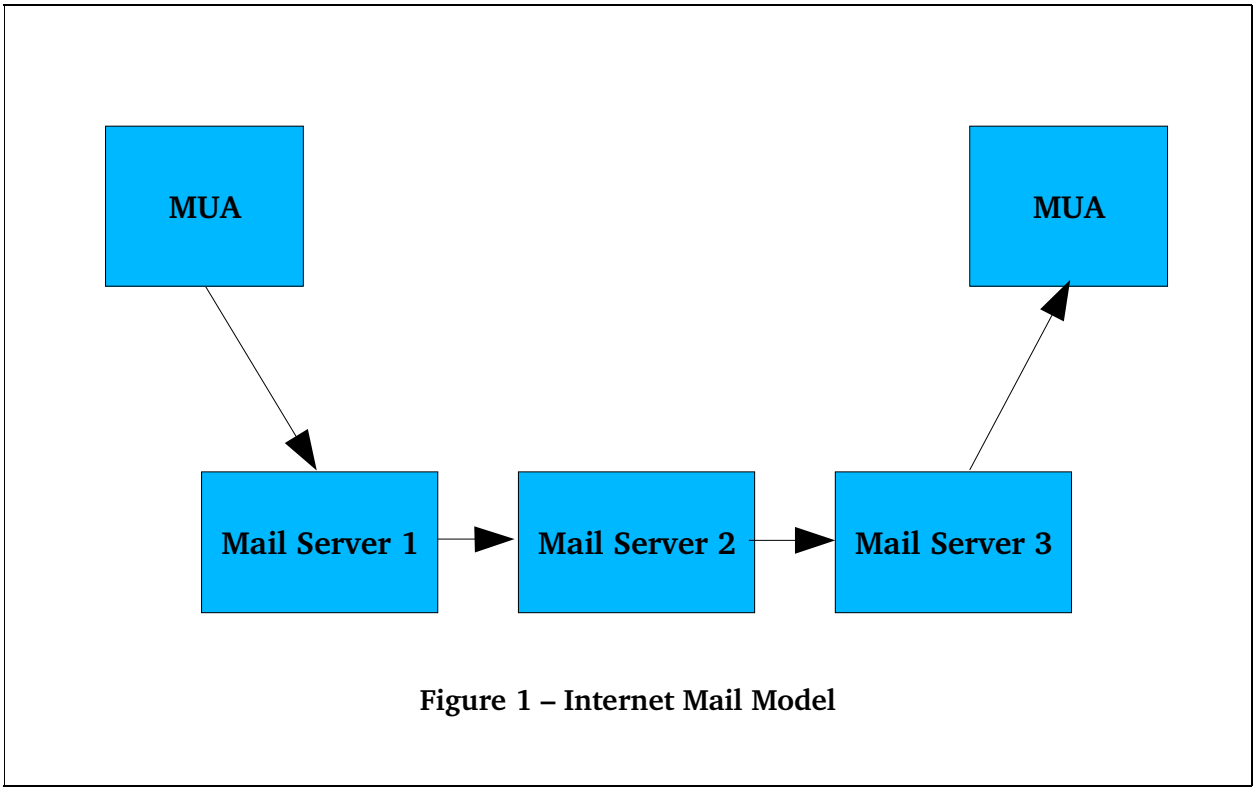


Figure 1 – Internet Mail Model

This diagram shows the path for the delivery of a single email. The mail is generated by a Mail User Agent (MUA). It is then passed to a mail server which has to decide whether it can deliver the mail locally or whether the mail must be passed to another server. The mail is passed from server to server until one of them decides that it can deliver the mail locally, which it then does. When this delivery is complete, the mail is then available to a MUA to read it. The final MUA has the responsibility of retrieving the mail as well passing it to a Mail User Interface (MUI) to display it to the user.

How each mail server decides whether to deliver locally or not could be the subject of another chapter. Briefly, each server consults a local configuration file or files together with information from DNS servers (principally the MX records). This is all then used to decide what is considered local. For non-local mail the server then uses this information to determine the address of the next mail server to send the mail to.

Each mail server in general has the structure shown in the Figure 2.

The Mail Transport Agent (MTA) accepts connections from other mail servers and MUAs via the Simple Mail Transport Protocol (SMTP). If the mail is not for local delivery it is then sent by the MTA to another server. If the mail is for local delivery it is passed to a Mail Delivery Agent (MDA). The MDA is responsible for storing the mail in the user's mailstore. The mailstore is simply a way of

storing data; for instance, a file, a series of separate files or even an SQL database. The precise storage structure is defined by whatever the MDA supports.

When a user wants to view her mail she uses a MUA which either retrieves the mail directly or contacts a server side component which retrieves the mail from the mailstore and passes it to the MUA. Such server side components do not fit into the traditional MTA/MDA/MUA model and we shall call them Mail Access Agents (MAA). This term is, however, not in common use.

The MUA communicates with a MAA using an open protocol which is usually either the Post Office Protocol (POP), or the Internet Mail Access Protocol (IMAP). The POP protocol normally deletes mail from the mailstore when it is passed to the client and IMAP normally leaves it there. The IMAP protocol also allows the MUA to alter the mailstore, for example by deleting mail or moving it from one directory to another.

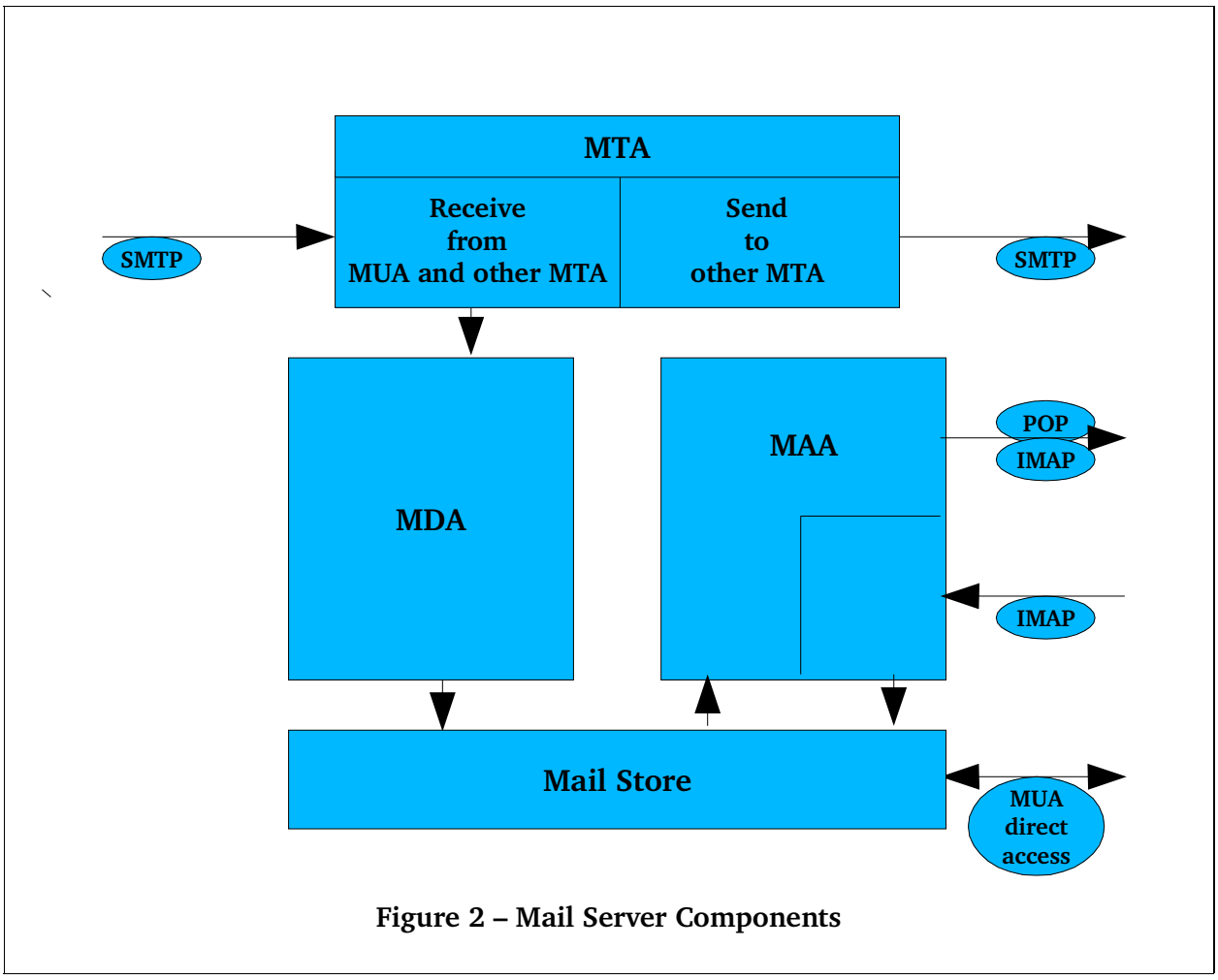


Figure 2 – Mail Server Components

The MUA may store the mail locally on the machine it is running on. This normally happens if POP is used. This local storage then allows future access to be independent of the server, which is particularly useful for machines that are not permanently connected to the network. IMAP, on the other hand, normally operates without local copies but it can also operate in what is called disconnected mode, which maintains a local copy, thereby allowing mail to be manipulated without a network connection. In this mode, the local and server mailstores are synchronised when a network connection is made. Unfortunately, not all MUAs fully support disconnected IMAP.

Sometimes a program other than a MUA retrieves the mail and stores it locally for a MUA to access without having to connect to the server itself. Such programs pull mail onto their machine in contrast to a standard MTA which has mail pushed to it by other MTAs. This can be useful if users do not want to allow connections to their machines from the Internet, or are operating behind a firewall. An example of such a program is *fetchmail*.

The difficulty with this model is that the applications available do not always map directly onto it. Applications very often do more than one of the functions; for instance a MTA may incorporate the MDA function, and the popular MTA *Sendmail* can even be used as a MUA in some circumstances.

As mail is passed from the originating MUA through the various servers to the final MUA, a series of headers is added which records details of the journey and also controls the processing of the mail by both the intervening servers and the final MUA. Some of these are Multi-purpose Internet Mail Extension (MIME) headers which are used for a range of control purposes, including support for non-ASCII character sets, support for embedded content such as images, and support for attachments. When a MUA attaches a file it records its type as a MIME header and it is then the responsibility of the final MUA to be able to decode it.

Parts of this model are discussed in more detail below.

C.1. MTA

Most MTAs allow the administrator to control where mail is to be accepted from. This is often done by limiting the range of IP addresses that the MTA will accept SMTP connections from. This is extremely valuable in preventing spammers using the MTA as a relay and clogging the network bandwidth to the MTA.

There is a set of about 20 extensions to SMTP called Extended SMTP or ESMTP. These extensions allow, among other things, faster transfer of mail between compliant MTAs, using the pipeline extension.

Another extension enables Transport Layer Security (TLS) encryption between compliant MTAs, and another, SMTP-AUTH, allows users to be authenticated using a range of techniques. Both extensions are useful when the MTA would not normally allow a client to connect because its IP address is outside its trusted address space. This could happen for instance if a laptop user dials in from a random site on the Internet. See C.4.2 below.

The original model assumed that the owner of a mail account had a login account on the mail server. This meant that the MTA could interrogate the local password file to authenticate users. This model is too restrictive and modern MTAs must now support Virtual Users where the account owner details are held in a database, often independently of the normal login account details. This means that a user could have a password for mail and a different one for logging in.

The database could be LDAP-supported, an SQL database or a flat file. *MySQL* is the preferred SQL server as it is efficient and fast in what is basically a read-only application. *PostgreSQL* and *Oracle* could also be used.

An LDAP-supported database is recommended as they offer better support for distribution.

Default LDAP implementations often use the Berkeley Database products from Sleepycat Systems.

Sometimes a machine may only connect to a mail server intermittently. This may happen in the case of home workers or laptop users, for instance. It may also happen in small offices where the cost of a permanent connection cannot be justified. In these circumstances the central MTA cannot forward mail as it would normally would, and it needs to store it until a connection is

made. Similar comments hold for the MTA (if there is one) on the client machine, or, in the case of a small office, the gateway mail server. These MTAs need to be able to support such situations and they are often called Smart Hosts when they do.

Delivery from a Smart Host may be carried out using SMTP or POP3.

Delivery via SMTP is straightforward and security on the receiving machine can be improved by restricting inward connection from the Smart Host only.

Delivery via POP3 is achieved either by using a MUA or the *fetchmail* application. *Fetchmail* will download mail to a local mailstore as mentioned above or deliver to a local MTA if required, for instance where multiple mail accounts are involved.

Both of these methods work well but have the drawback that they do not allow the use of blacklists to prevent spam from open relays and other undesirable sources being accepted. Tools like *SpamAssassin* can eliminate most of the spam, but the processing costs are much higher, and increased bandwidth is used downloading the mail for examination.

C.2. MUA

The MUA and MUI together make up the package that most users think of as “the mailer”. This is the client software that runs either on a web server or directly on a desktop machine to allow people to send and receive mail. Some sort of storage is normally provided so that mail can be filed into “folders” or “local mailboxes” for future reference.

The MUA handles protocols such as SMTP for mail submission and IMAP or POP for mail retrieval and filing. It understands the format of mail messages and can decompose MIME messages into their component parts.

Where there is a requirement for strong end-to-end security, the MUA is also responsible for the encryption and signing of messages. There are two competing standards for this: S/MIME, which is based on X.509 certificates, and PGP/GPG which is based on a different certificate format with a web-of-trust model rather than a hierarchy-of-trust.

Most of the OSS MUAs support digital signing using the *GNU Privacy Guard* (*GPG*). Only a few support S/MIME signatures. Business and Government bodies have opted for the S/MIME standard and its use must therefore be supported.

C.3. Mailstore

Unix mail systems originally assumed that the owner of a mail account had access to the machine hosting the mail server and could read a file containing their mail – or, alternately, that mail would be delivered to the machine the user usually logged into to work. This was fine for environments with a small number of users who also needed a real login account on a machine with a mail server, but it is not practicable or secure generally.

The original format for storing mail was a single file per user with new mail being appended to the end. This file could get very large and reading through it to read a random piece of mail soon became inefficient. This format is often known as “mbox” and is still used by some MUAs, in particular for mail stored locally for the user. A modification to this was to hold each piece of mail as a separate file in a directory structure, which allows more efficient random access. One variant of this structure is called “mh” and a particular one with certain defined sub-directories and access procedures is called “maildir”.

Sometimes these structures were held on the mail server and exported to the clients using, for instance, NFS. This allowed the mail to be held centrally, which meant that it could be backed

up properly, but introduced locking problems with the single file structure. However, using NFS has not proved popular, perhaps due to the lack of good NFS clients on *Windows*.

Not all MTAs support these different access methods directly, thus the need for MAAs. A MUA which cannot access the mailstore directly will have to use a MAA component using POP or IMAP.

Both POP3 and IMAP send passwords as clear text by default. IMAP can use hashed passwords if the MUA supports them. Use of TLS encrypted links is possible if the MAA and MUA support it, is advisable on local networks, and should be mandatory for remote access.

MTAs sometimes communicate with MDAs using the Local Mail Transport Protocol or LMTP. Most of the MTAs and MDAs support this.

C.4. Roaming Users

The problem with roaming users is that they can connect from unpredictable IP addresses, so the normal methods used by MTAs to decide whether to accept incoming mail will prevent them sending mail via the Administration's mail server. The MTAs have to restrict access to themselves from unknown clients to prevent their use by spammers as a third-party relay.

Three general techniques are available to get round this problem.

C.4.1. Virtual Private Networks (VPNs)

In a VPN, the remote machine can be allocated an address that can be included in the MTAs trusted address space. The problem is that full access to the internal network will be available to anyone who gets access to the remote machine, a significant risk with laptops, unless the access keys are encrypted with a password entered whenever the connection is initiated. Unfortunately, users sometimes configure their machines to remember passwords.

C.4.2. SMTP-AUTH and TLS

The SMTP-AUTH extension to SMTP allows a MTA to be configured to request a password to authenticate the remote user. The principal authentication methods are PLAIN, LOGIN and CRAM-MD5.

PLAIN requires the password to be held in clear on the client but can be encrypted on the server. If the SMTP connection is unencrypted then the password is passed in clear (although in base-64) over the network.

LOGIN is less efficient than PLAIN as it requires three network interactions rather than one and, like PLAIN, the username and password travel in clear over the network.

CRAM-MD5 encrypts the username and password as they pass over the network. However, the password must be held in clear text on both the client and server. It requires only two network interactions.

Not all MUAs support SMTP-AUTH and those that do may only support a limited number of methods. For instance *Outlook Express* only uses LOGIN.

Compared to using a VPN, the only access enabled is sending mail, so other services would not be compromised if the remote machine were stolen.

ESMTP also allows a TLS session to be negotiated between the client and server. This connection encrypts data on the network and can also authenticate the client machine. Authentication requires a client certificate matching one held on the server.

C.4.3. POP-before-SMTP.

This method takes advantage of the fact that the POP and IMAP protocols require password authentication.

After a successful POP or IMAP connection to read mail, the MAA maintains an authenticated login database with the client IP address, date and time. When the client tries to send mail that is not for the local domain, the MTA checks whether the client IP address is in its trusted address space. If not, it then checks the authenticated login database for the IP address. If no authenticated login has been recorded from the client's IP address, or the last authenticated connection didn't happen recently enough, the MTA refuses to relay the message. The time period is configurable and typically defaults to 20 minutes. This method needs the MAA and the MTA to co-operate. For this reason, not all combinations work.

This method has the disadvantage that users must check for incoming mail first. Some users may find this difficult unless the MTA automatically does it for them.

C.5. Performance

In general an MTA uses little processor power; hosts running nothing else are usually limited by the network bandwidth or disk performance. IMAP and POP servers require more processor power and IMAP requires a little more RAM than POP. However, none of this is likely to be an issue on current hardware.

Anti-virus scanners require a lot of RAM and processor power, especially if MIME attachments are allowed.

Even so, performance limits are usually set by traffic rather than by the number of accounts.

Below are some performance examples culled from reports to mailing lists and from the experience of **netproject**'s consultants. These are included to give some idea of what is required:

Site 1 – 2 x *Pentium III Xeon* 2.4G, 4 Gb RAM, 3 x 36GB SCSI Raid 5

Virtual Users with lookup on *MySQL*.

Postfix 2.0.6, *Courier-IMAP* 1.7, *MySQL* 4.12, *RAV-Antivirus*, *Mailman* 2.1, *Red Hat Linux 8.0*, no SSL.

Around 4,800 users.

Site 2 – *Athlon* 1200, 1 Gb RAM, RAID5

Postfix + *Courier-IMAP* (anti-virus scan on another machine), no SSL.

8,500 Users.

Site 3 – *Pentium* 133, 40 Mb RAM, IDE disk

Debian GNU/Linux, *Courier-MTA* + *Courier-IMAP* + *SpamAssassin* (the latter for one user only).

Typically 18 POP3 users and 7 IMAP users at any time.

Processor about 20% occupied.

Site 4 – dual *Pentium II* 450 *Xeon*, 256 Mb RAM

MySQL, *Courier-MTA*, *Courier-IMAP*, *sqwebmail*, SSL.

50 users, mainly POP3.

Site 5 – *Pentium II* 400 with 256M RAM

Courier-MTA + *SpamAssassin*, *Red Hat Linux* 8.0.

300 mailboxes, around 4,000 messages per day.

Site 6 – *Pentium III* 677Mhz, 512Mb RAM, 2 x IDE disk

FreeBSD 4.7, *Exim* 4.05, *OpenLDAP* 2.1.5, *Cyrus* 2.1.11, *Mailman* 2.1, *Apache* 1.3.26.

The machine is principally a busy webserver, but it also handles several thousand mails per day without any noticeable additional load.

Appendix D. Desktop Reference Software

The list shows the RPM packages and their version numbers. The list is based on *Red Hat Linux* version 8.0 together with Ximian's *Red Carpet* update software, and includes only the minimal set of dependencies for the target environment. *Evolution* has been updated through *Red Carpet* to a later release than that delivered with *Red Hat*.

NAME	*VERSION*	*RELEASE*
anacron	2.3	23
aspell	0.33.7.1	ximian.4
atk	1.0.3	1
audiofile	0.2.3	3
basesystem	8	1
bash	2.05b	5
bdflush	1.5	21
bitmap-fonts	0.2	2
bonobo	1.0.21	1.ximian.1
bonobo-activation	1.0.3	2
bonobo-conf	0.16	1.ximian.1
bzip2-libs	1.0.2	5
chkconfig	01/03/06	3
chkfontpath	01/09/06	3
compat-libstdc++	7.3	2.96.110
control-center	2.0.1	8
cpp	3.2	7
cracklib	2.7	18
cracklib-dicts	2.7	18
crontabs	1.1	4
cups-libs	01/01/17	0.2
cyrus-sasl	02/01/10	1
cyrus-sasl-md5	02/01/10	1
db4	4.0.14	14
desktop-backgrounds-basic	2	10
desktop-backgrounds-extra	2	10
desktop-file-utils	0.3	3
dev	03/03/01	2
dhclient	3.0pl1	26

NAME	*VERSION*	*RELEASE*
dialog	0.9b	20020519.1
diffutils	02/08/01	3
docbook-dtds	1	14
e2fsprogs	1.27	9
ed	0.2	28
eel2	2.0.6	1
emacs	21.2	18
eog	1.0.2	5
esound	0.2.28	1
evolution	01/02/02	1.ximian.1
expat	1.95.4	1
fam	02/06/08	4
filesystem	02/01/06	5
fileutils	04/01/09	11
findutils	04/01/07	7
fontconfig	2	3
fortune-mod	1	24
freetype	02/01/02	7
gail	0.17	2
galeon	01/02/06	0.8.0
gawk	03/01/01	4
Gconf	1.0.9	6
Gconf2	01/02/01	3
gdbm	01/08/00	18
gdk-pixbuf	0.18.0	4
gdk-pixbuf-gnome	0.18.0	4
gdm	2.4.0.7	13
gedit	2.0.2	5
gftp	2.0.13	5
ghostscript	7.05	20
ghostscript-fonts	5.5	7
gimp	01/02/03	9
gimp-print	04/02/01	5
glib	01/02/10	8

NAME	*VERSION*	*RELEASE*
glib2	2.0.6	2
glibc	02/03/02	4.80.6
glibc-common	02/03/02	4.80.6
Glide3	20010520	19
gmp	4.1	4
gnome-desktop	2.0.6	4
gnome-libs	1.4.1.2.90	22
gnome-mime-data	2.0.0	9
gnome-panel	2.0.6	9
gnome-session	2.0.5	7
gnome-spell	0.5	1.ximian.3
gnome-terminal	2.0.1	5
gnome-utils	2.0.2	5
gnome-vfs	1.0.5	6.ximian.1
gnome-vfs2	2.0.2	5
gnupg	1.0.7	6
gpm	1.19.3	23
grep	02/05/01	4
grub	0.92	7
gtk+	01/02/10	22
gtk2	2.0.6	8
gtkhtml1.1	01/01/08	1.ximian.1
gzip	01/03/03	5
htmlview	2.0.0	6
hwdata	0.48	1
imlib	01/09/13	9
indexhtml	8	1
info	4.2	5
initscripts	6.95	1
intltool	0.22	3
iproute	02/04/07	5
iputils	20020124	8
kbd	1.06	26
krb5-libs	01/02/05	15

NAME	*VERSION*	*RELEASE*
krbafs	01/01/01	6
less	358	28
libacl	2.0.11	2
libart_lgpl	02/03/10	1
libattr	2.0.8	3
libbonobo	2.0.0	4
libbonobo-conf0	0.16	1.ximian.1
libbonoboui	2.0.1	2
libcapplet0	1.4.0.1	9
libelf	0.8.2	2
libgal21	0.23	1.80.ximian.1
libgcc	3.2	7
libghttp	1.0.9	5
libglade	0.17	8
libglade2	2.0.0	2
libgnome	2.0.2	5
libgnomecanvas	2.0.2	1
libgnomeprint	1.116.0	2
libgnomeprint15	0.37	2.ximian.1
libgnomeprintui	1.116.0	1
libgnomeui	2.0.3	3
libgtkhtml1.1-3	01/01/08	1.ximian.1
libjpeg	6b	21
libmng	1.0.4	1
libpng10	1.0.13	6
libpng	01/02/02	8
librpm404	4.0.4	8x.27
librsvg2	2.0.1	1
libstdc++	3.2	7
libtermcap	2.0.8	31
libtiff	03/05/07	7
libungif	04/01/00	13
libuser	0.51.1	2
libwnck	0.17	1

NAME	*VERSION*	*RELEASE*
libxml	01/08/17	5
libxml2	02/04/23	1
libxslt	1.0.19	1
linc	0.5.2	2
logrotate	03/06/05	2
losetup	2.11r	10
lvm	1.0.3	9
metacity	2.4.0.92	5
mingetty	1	3
mkinitrd	03/04/28	1
mktemp	1.5	16
modutils	02/04/18	2
mount	2.11r	10
mozilla	1.0.1	26
mozilla-nspr	1.0.1	26
mozilla-nss	1.0.1	26
mozilla-psm	1.0.1	26
nautilus	2.0.6	6
ncurses	5.2	28
net-tools	1.6	7
newt	0.51.0	1
nfs-utils	1.0.1	2
nscd	02/03/02	4.80.6
nss_ldap	198	3
oaf	0.6.10	1.ximian.2
Omni	0.7.0	6
openjade	01/03/01	9
openldap	2.0.27	02/08/00
openoffice	1.0.1	8
openoffice-i18n	1.0.1	8
openoffice-libs	1.0.1	8
openssh	3.4p1	2
openssh-clients	3.4p1	2
openssh-server	3.4p1	2

NAME	*VERSION*	*RELEASE*
openssl	0.9.6b	33
ORBit	0.5.13	5
ORBit2	02/04/01	1
pam	0.75	46.8.0
pango	01/01/01	1
passwd	0.67	3
patch	02/05/04	14
pcre	3.9	5
perl	05/08/00	55
perl-Filter	1.28	9
popt	1.7	1.06
portmap	4	46
procps	2.0.7	25
psmisc	20.2	6
pspell	0.12.2	ximian.7
python	02/02/01	17
qt	3.0.5	17
rc	01/02/01	1.ximian.1
rcd	01/02/01	1.ximian.3
readline	4.3	3
redhat-artwork	0.47	3
redhat-logos	01/01/06	2
redhat-menus	0.26	1
redhat-release	8	8
rootfiles	7.2	4
rpm	4.1	1.06
scrollkeeper	0.3.10	7
sed	3.02	13
setup	02/05/20	1
sgml-common	0.6.3	12
shadow-utils	20000902	12.8
sh-utils	2.0.12	3
slang	01/04/05	11
soup	0.7.11	1.ximian.1

NAME	*VERSION*	*RELEASE*
switchdesk	03/09/08	9
sysklogd	01/04/01	10
SysVinit	2.84	5
tar	1.13.25	8
termcap	11.0.1	13
tetex	1.0.7	57.1
tetex-fonts	1.0.7	57
textutils	2.0.21	5
tmpwatch	02/08/04	3
urw-fonts	2	26
usermode	1.63	1
utempter	0.5.2	10
util-linux	2.11r	10
Vflib2	2.25.6	8
vixie-cron	3.0.1	69
vnc	3.3.3r2	39.2
vnc-doc	3.3.3r2	39.2
vte	0.8.19	2
which	2.14	1
words	2	20
Xaw3d	1.5	16
Xfree86	04/02/00	72
Xfree86-75dpi-fonts	04/02/00	72
Xfree86-base-fonts	04/02/00	72
Xfree86-font-utils	04/02/00	72
Xfree86-libs	04/02/00	72
Xfree86-Mesa-libGL	04/02/00	72
Xfree86-Mesa-libGLU	04/02/00	72
Xfree86-truetype-fonts	04/02/00	72
Xfree86-xauth	04/02/00	72
Xfree86-xfs	04/02/00	72
Xft	2	1
xinetd	02/03/07	5
xinitrc	3.31	1

NAME	*VERSION*	*RELEASE*
xml-common	0.6.3	12
xpdf	1.01	10
xscreensaver	4.05	6
xsri	02/01/00	3
zlib	01/01/04	8.8x

Appendix E. Server Reference Software

The list shows the RPM packages and their version numbers. The list is based on *Red Hat Linux* version 8.0 together with Ximian's *Red Carpet* update software. Note that this is not a minimal list. There are applications here which were left on for convenience, for instance the *Gnome* and *X*-related ones. These allowed the use of graphical management interfaces, but simple text-based interfaces could have been used instead.

NAME	*VERSION*	*RELEASE*
a2ps	4.13b	24
acl	2.0.11	2
adjtimex	1.13	4
alchemist	1.0.24	4
amanda	2.4.2p2	9
anaconda-help	8	1
anacron	2.3	23
apmd	3.0.2	12
ash	0.3.8	5
at	03/01/08	31
atk	1.0.3	1
attr	2.0.8	3
audiofile	0.2.3	3
authconfig	04/02/12	3
autoconvert	0.3.7	8
autofs	03/01/07	33
autorun	3.3	3
basesystem	8	1
bash	2.05b	5.1
bash-doc	2.05b	5.1
bdflush	1.5	21
beecrypt	02/02/00	6
bg5ps	01/03/00	9
bind	09/02/01	9
bind-utils	09/02/01	9
bitmap-fonts	0.2	2
blas-man	3	18
bonobo-activation	1.0.3	2
booty	0.12	1

NAME	*VERSION*	*RELEASE*
bridge-utils	0.9.3	6
bzip2	1.0.2	5
bzip2-libs	1.0.2	5
caching-nameserver	7.2	4
cadaver	0.20.5	2
cdrdao	01/01/05	10
chkconfig	01/03/06	3
chkfontpath	01/09/06	3
cleanfeed	0.95.7b	17
comps-extras	8	3
courier-imap	01/07/01	2.whoson.8.0
courier-imap-ldap	01/07/01	2.whoson.8.0
cpio	02/04/02	28
cracklib	2.7	18
cracklib-dicts	2.7	18
crontabs	1.1	4
cups-libs	01/01/17	0.7
curl	07/09/08	1
cWnn-common	1.11	27
cyrus-sasl	02/01/10	1
cyrus-sasl-gssapi	02/01/10	1
cyrus-sasl-md5	02/01/10	1
cyrus-sasl-plain	02/01/10	1
db4	4.0.14	14
db4-java	4.0.14	14
dev	03/03/01	2
dhclient	3.0pl1	26
dhcp	3.0pl1	26
dialog	0.9b	20020519.1
dictd	01/05/05	3
diffutils	02/08/01	3
diskcheck	1.3	2
docbook-dtds	1	14
dos2unix	3.1	12

NAME	*VERSION*	*RELEASE*
dosfstools	2.8	3
dtach	0.5	5
e2fsprogs	1.27	9
ed	0.2	28
eject	2.0.12	7
elinks	0.3.2	1
enscript	01/06/01	22
esound	0.2.28	1
ethtool	1.6	2
exim	4.12	4.rh8x
exim-ldap	4.12	4.rh8x
expat	1.95.4	1
fam	02/06/08	4
fbset	2.1	11
fetchmail	05/09/00	21/08/00
file	3.39	9
filesystem	02/01/06	5
fileutils	04/01/09	11
findutils	04/01/07	7
finger	0.17	14
fontconfig	2	3
freetype	02/01/02	7
FreeWnn-common	1.11	27
FreeWnn-libs	1.11	27
ftp	0.17	15
ftpcopy	0.5.1	1
gail	0.17	2
gawk	03/01/01	4
GConf2	01/02/01	3
gd	01/08/04	9
gdbm	01/08/00	18
genromfs	0.3	12
glib	01/02/10	8
glib2	2.0.6	2

NAME	*VERSION*	*RELEASE*
glibc	02/03/02	4.80.6
glibc-common	02/03/02	4.80.6
gmp	4.1	4
gnome-audio-extra	01/04/00	4
gnome-mime-data	2.0.0	9
gnome-python2	1.99.11	8
gnome-python2-bonobo	1.99.11	8
gnome-python2-canvas	1.99.11	8
gnome-python2-gtkhtml2	1.99.11	8
gnome-vfs2	2.0.2	5
gnupg	1.0.7	8
gpg-pubkey	db42a60e	37ea5438
gpm	1.19.3	23
grep	02/05/01	4
groff	1.18	6
groff-perl	1.18	6
grub	0.92	7
gsl	01/01/01	3
gtk+	01/02/10	22
gtk2	2.0.6	8
gtkhtml2	2.0.1	2
gzip	01/03/03	5
h2ps	2.06	6
hdparm	5.2	1
hesiod	3.0.2	21
hotplug	2002_04_01	13
htmlview	2.0.0	6
httpd	2.0.40	11.5
hwdata	0.48	1
imlib	01/09/13	9
indexhtml	8	1
inews	02/03/03	5
info	4.2	5
initscripts	6.95	1

NAME	*VERSION*	*RELEASE*
intltool	0.22	3
iproute	02/04/07	5
iptables	1.2.6a	2
iptraf	02/07/00	3
iputils	20020124	8
ipxutils	2.2.0.18	11
isicom	3.05	6
jcode.pl	2.13	6
jfsutils	1.0.17	3
kakasi	02/03/04	8
kakasi-dict	02/03/04	8
kbd	1.06	26
kbdconfig	01/09/16	1
kcc	2.3	14
kdoc	3.0.0	2.cvs20020321.3
kernel	02/04/20	13.8
kernel	02/04/20	18.8
kernel-utils	2.4	8.28
krb5-libs	01/02/05	15
krbafs	01/01/01	6
krbafs-utils	01/01/01	6
ksymoops	02/04/05	1
kudzu	0.99.69	1
lapack-man	3	18
less	358	28
lftp	02/05/02	5
libacl	2.0.11	2
libaio	0.3.13	5
libao	0.8.3	1
libart_lgpl	02/03/10	1
libattr	2.0.8	3
libbonobo	2.0.0	4
libbonoboui	2.0.1	2
libcap	1.1	12

NAME	*VERSION*	*RELEASE*
libelf	0.8.2	2
libesmtp	0.8.12	2
libf2c	3.2	7
libgcc	3.2	7
libghttp	1.0.9	5
libglade2	2.0.0	2
libgnome	2.0.2	5
libgnomecanvas	2.0.2	1
libgnomeui	2.0.3	3
libgtop	1.0.12	11
libIDL	0.8.0	3
libjpeg	6b	21
libmng	1.0.4	1
libobjc	3.2	7
libogg	1	1
libole2	0.2.4	4
libpng10	1.0.13	6
libpng	01/02/02	8
librpm404	4.0.4	8x.27
libstdc++	3.2	7
libtermcap	2.0.8	31
libtiff	03/05/07	7
libtool-libs	01/04/02	12
libungif	04/01/00	13
libunicode	0.4	9
libusb	0.1.6	1
libuser	0.51.1	2
libwvstreams	3.7	5
libxml10	1.0.0	11
libxml2	02/04/23	1
libxml2-python	02/04/23	1
libxslt	1.0.19	1
lilo	21/04/04	20
linc	0.5.2	2

NAME	*VERSION*	*RELEASE*
lm_sensors	02/06/03	2
lockdev	1.0.0	20
logrotate	03/06/05	2
logwatch	2.6	8
lokkit	0.5	21/08/00
losetup	2.11r	10
LPRng	03/08/09	6.1
lvm	1.0.3	9
m2crypto	0.05_snap4	6
m4	01/04/01	11
macutils	2.0b3	22
mailcap	02/01/12	1
mailman	2.0.13	3
mailx	08/01/01	26
make	3.79.1	14
man	1.5k	0.8x.0
man-pages	1.53	1
mc	04/05/55	12
mdadm	1.0.0	6
mew-common	2.2	6
mingetty	1	3
mkbootdisk	01/04/08	1
mkinitrd	03/04/28	1
mkisofs	1.1	14
mktemp	1.5	16
mod_perl	1.99_05	3
mod_python	3.0.0	10
modutils	02/04/18	2
mount	2.11r	10
mouseconfig	4.26	1
mozilla-nspr	1.0.1	26
mozilla-nss	1.0.1	26
mpage	02/05/02	4
mtools	03/09/08	5

NAME	*VERSION*	*RELEASE*
mtr	0.49	7
mt-st	0.7	6
mtx	01/02/16	5
namazu	2.0.10	8
namazu-cgi	2.0.10	8
nc	1.1	16
ncftp	03/01/03	6
ncompress	04/02/04	31
ncpfs	2.2.0.18	11
ncurses4	5	9
ncurses	5.2	28
netconfig	0.8.12	3
netpbm	9.24	9.80.2
net-snmp	5.0.6	8.80.2
net-snmp-utils	5.0.6	8.80.2
net-tools	1.6	7
newt	0.51.0	1
nfs-utils	1.0.1	2
nhpf	1.42	3
nkf	1.92	11
nmap	3	1
nscd	02/03/02	4.80.6
nss_ldap	198	3
ntp	4.1.1a	9
ntsysv	01/03/06	3
nut-cgi	0.45.4	5
open	1.4	16
openjade	01/03/01	9
openldap12	01/02/13	9
openldap	2.0.27	02/08/00
openldap-clients	2.0.27	02/08/00
openldap-servers	2.0.27	02/08/00
openssh	3.4p1	2
openssh-clients	3.4p1	2

NAME	*VERSION*	*RELEASE*
openssh-server	3.4p1	2
openssl095a	0.9.5a	21
openssl096	0.9.6	16.8
openssl	0.9.6b	33
openssl-perl	0.9.6b	33
ORBit	0.5.13	5
ORBit2	02/04/01	1
orbit-python	1.99.0	4
pam	0.75	46.8.0
pam_krb5	1.56	1
pam_smb	01/01/06	5
pango	01/01/01	1
parted	01/04/24	6
passwd	0.67	3
patch	02/05/04	14
pciutils	02/01/10	2
pcre	3.9	5
pdksh	05/02/14	19
perl	05/08/00	55
perl-Crypt-SSLeay	0.45	2
perl-DateManip	5.4	27
perl-Digest-SHA1	2.01	6
perl-File-MMagic	1.15	2
perl-Filter	1.28	9
perl-Frontier-RPC	0.06	33
perl-HTML-Parser	3.26	14
perl-HTML-Tagset	3.03	25
perl-libwww-perl	5.65	2
perl-libxml-enno	1.02	25
perl-libxml-perl	0.07	25
perl-Mail-SpamAssassin	2.53	1
perl-NKF	1.71	7
perl-Parse-Yapp	1.05	26
perl-Text-Kakasi	1.05	2

NAME	*VERSION*	*RELEASE*
perl-TimeDate	1.13	2
perl-Time-HiRes	1.2	23
perl-URI	1.21	3
perl-XML-Dumper	0.4	22
perl-XML-Encoding	1.01	20
perl-XML-Grove	0.46alpha	21
perl-XML-Parser	2.31	12
pinfo	0.6.4	7
pnm2ppa	1.04	5
popt	01/07/01	1.8x
portmap	4	46
ppp	02/04/01	7
prelink	0.2.0	8
procinfo	18	5
procmail	3.22	7
procps	2.0.7	25
psmisc	20.2	6
pspell	0.12.2	14
psutils	1.17	17
pwlib	01/03/03	5
pygtk2	1.99.12	7
pygtk2-libglade	1.99.12	7
pyOpenSSL	0.5.0.91	1
python	02/02/01	17
python-optik	1.3	2
pyxf86config	0.3.1	2
quota	3.06	5
raidtools	1.00.2	3.3
rc	01/04/01	0.ximian.5.1
rcd	01/04/03	0.ximian.5.1
rdate	1.2	5
rdist	06/01/05	24
readline41	4.1	14
readline	4.3	3

NAME	*VERSION*	*RELEASE*
recode	3.6	6
red-carpet	2.0.1	0.ximian.5.1.1
redhat-config-packages	1.0.1	1
redhat-logos	01/01/06	2
redhat-menus	0.26	1
redhat-release	8	8
redhat-switchmail	0.5.14	1
redhat-switch-printer	0.5.12	1
reiserfs-utils	03/06/02	2
rhmask	1.2	2
rhn-applet	2.0.9	0.8.0.1
rhnlib	1	1
rhpl	0.51	1
rootfiles	7.2	4
rpm404-python	4.0.4	8x.27
rpm	04/01/01	1.8x
rpm-python	04/01/01	1.8x
rp-pppoe	3.4	7
rsh	0.17	10
ruby-docs	01/06/07	10
samba	02/02/07	05/08/00
samba-client	02/02/07	05/08/00
samba-common	02/02/07	05/08/00
sane-backends	1.0.8	5
sash	3.4	14
screen	03/09/11	10
scrollkeeper	0.3.10	7
sed	3.02	13
setserial	2.17	9
setup	02/05/20	1
setuptool	1.1	1
sgml-common	0.6.3	12
shadow-utils	20000902	12.8
shapecfg	02/02/12	10

NAME	*VERSION*	*RELEASE*
sh-utils	2.0.12	3
slang	01/04/05	11
slocate	2.6	4
spamassassin	2.53	1
spamassassin-tools	2.53	1
stat	3.3	4
statserial	1.1	30
sudo	01/06/06	1
symlinks	1.2	16
sysklogd	01/04/01	10
syslinux	1.75	3
sysstat	4.0.5	3
SysVinit	2.84	5
talk	0.17	17
tar	1.13.25	8
tcp_wrappers	7.6	23
tcsh	6.12	2
telnet	0.17	23
termcap	11.0.1	13
textutils	2.0.21	5
tftp	0.29	3
time	1.7	19
timeconfig	03/02/09	1
tmpwatch	02/08/04	3
traceroute	1.4a12	6
tree	1.2	20
ttcp	1.12	5
ttfprint	0.9	6
unix2dos	2.2	17
up2date	3.0.7	1
up2date-gnome	3.0.7	1
usbutils	0.9	7
usermode	1.63	1
usermode-gtk	1.63	1

NAME	*VERSION*	*RELEASE*
utempter	0.5.2	10
util-linux	2.11r	10
vim-common	6.1	18.8x.1
vim-enhanced	6.1	18.8x.1
vim-minimal	6.1	18.8x.1
vixie-cron	3.0.1	69
vlock	1.3	11
vnc-doc	3.3.3r2	39.2
w3c-libwww	05/04/00	1
w3c-libwww-apps	05/04/00	1
w3m-el-common	01/03/01	1
watanabe-vf	1	8
webalizer	2.01_10	9
wget	01/08/02	5
which	2.14	1
whois	1.0.10	4
whoson	2.02a	1
wireless-tools	25	1
wl-common	02/08/01	8
Wnn6-SDK	1	21
words	2	20
wvdial	1.53	7
xferstats	2.16	3
XFree86-base-fonts	04/02/00	72
XFree86-doc	04/02/00	72
XFree86-font-utils	04/02/00	72
XFree86-libs	04/02/00	72
XFree86-Mesa-libGL	04/02/00	72
XFree86-Mesa-libGLU	04/02/00	72
XFree86-xfs	04/02/00	72
Xft	2	1
xinetd	02/03/11	01/08/00
xml-common	0.6.3	12
ypbind	1.11	2

NAME	*VERSION*	*RELEASE*
yp-tools	2.7	3
zisofs-tools	1.0.3	5
zlib	01/01/04	8.8x
zsh	4.0.4	8

Appendix F. Desktop Installation Code

This code is used to set up a desktop with French as the default language. It uses Red Hat's **kickstart** program to install and Ximian's *Red Carpet* to update. It is also tailored for a specific hardware setup and machine configuration, including a particular video card and network configuration; changes would have to be made to cater for the Administration's particular setup. The code shown here is an example only, to show that the functionality can be built, and although it should work, is not guaranteed in any way by **netproject**.

```
lang fr_FR
langsupport fr_FR

# Other languages may need other keyboards
keyboard uk

# This defines the type of mouse attached to the client.
mouse --emulthree genericps/2

# This needs changing to the location of the desktop.
timezone --utc Europe/London

# The password shown here would need changing
rootpw --iscrypted $1$7QNhVztt$2/DrxHONbGs91.D5k4rx21
reboot
install

# Change 192.168.1.1 to the IP address of the server.
# Change /mnt/space/RedHat-8.0 to the location on the server of the
# software repository.
nfs --server 192.168.1.1 --dir /mnt/space/RedHat-8.0
bootloader --location=mbr --append vga=0x305
zerombr yes
clearpart --linux --initlabel
part / --fstype ext3 --size 256 --grow --maxsize 512
part /usr --fstype ext3 --size 927 --grow --maxsize 2048
part swap --recommended
network --bootproto dhcp

# Change 192.168.1.1 to the servers IP address
auth --useshadow --enablemd5 --enableldap --enableldapauth --ldapserver
192.168.1.1 --ldapbasedn dc=netproject,dc=com
firewall -disabled

# This might need changing to reflect the monitor being used.
xconfig --depth 16 --resolution 1024x768 --defaultdesktop=GNOME --
startxonboot --card "S3 Savage (generic)" --videoram 16384

%packages --resolvedeps
librpm404
redhat-artwork
gnome-session
XFree86
gdm
openoffice-libs
openoffice-i18n
libglade
gdk-pixbuf
GConf
compat-libstdc++
indexhtml
```

```
libcapplet0
gdk-pixbuf-gnome
libghttp
mozilla-nss
metacity
nautilus
gnome-panel
control-center
XFree86-75dpi-font
gftp
gedit
emacs
desktop-backgrounds-extra

%pre --interpreter /usr/bin/python
import os, fcntl, CDROM
def eject(file):
    try:
        f = os.open(file, os.O_RDONLY | os.O_NONBLOCK)
    except OSError, (errno, strerror):
        print "%s - OS error(%s): %s" % (file, errno, strerror)
        return
    while 1:
        try:
            fcntl.ioctl(f, CDROM.CDROMEJECT);
        except IOError, (errno, strerror):
            print "%s - I/O error(%s): %s" % (file, errno, strerror)
            if (errno == 16):
                continue
        break
    os.close(f)
eject("/dev/hda")
eject("/dev/hdb")
eject("/dev/hdc")
eject("/dev/hdd")

%post
/bin/sh << "EOF" >> /root/preboot.log 2>&1
MNT=/mnt/tmp

# Change 192.168.1.1 to the servers IP address
HOST=192.168.1.1
date
mkdir -p $MNT
mount $HOST:/kickstart $MNT -t nfs -o ro
cd $MNT
./postinst.sh preboot
EOF
```

The code above calls a script called **postinst.sh** which contains the following:

```
#!/bin/sh
# description: Installation & upgrade script
# chkconfig: 2345 25 03
# © netproject 2003
# Sean Atkinson <sean@netproject.com>, March 2003

SERVICE=postinst

case $1 in
```

```
    start)
      exec $0 postboot >> /root/postboot.log 2>&1
      ;;
    stop)
      exit 0
      ;;
    preboot)
      rpm -U *.rpm
      mkdir /kickstart/
      cp -v diffs.patch gdm_bg.png gdm_logo.png /kickstart/
      cp -v grub_splash.xpm.gz /boot/grub/
      cp -v postinst.sh /etc/rc.d/init.d/$SERVICE
      chkconfig --add $SERVICE
      exit 0
      ;;
    postboot)
      ;;
    *)
      echo "Usage: $0 start|stop|preboot|postboot"
      exit 1
      ;;
esac

# Change 192,168.1.1 to servers IP address. Also details need to reflect
# the actual location and version of the distribution used
HOST=192.168.1.1
RCD=http://$HOST/rcd/
REL=redhat-80-i386

date

# The rpm commands below refer to specific versions of packages. the
# latest versions should be substituted if needed.
rpm -U $RCD/redcarpet/$REL/rcd-1.2.1-1.ximian.3.i386.rpm
cat << EOF >> /etc/ximian/rcd.conf
[Network]
host=$RCD

[Cache]
enabled=false
EOF
service rcd start
rpm -U $RCD/redcarpet/$REL/rc-1.2.1-1.ximian.1.i386.rpm
until rc ping
  do
  sleep 1
done
rc subscribe $REL redcarpet ximian-evolution
service rcd restart
rpm -e kernel-pcmcia-cs
rpm -e kudzu
rpm -e comps
rpm -e authconfig
until rc ping
  do
  sleep 1
done
rc update --no-confirmation
for PKG in galeon evolution xpdf vnc vnc-doc openoffice
  do
  rc install --no-confirmation $PKG
done
```

```
# This command assumes a specific release of the java run time
# environemnt
ln -s /usr/java/j2re1.4.1_02/plugin/i386/ns610/libjavaplugin_oji.so \
    /usr/lib/mozilla-1.0.1/plugins/
patch -p0 < /kickstart/diffs.patch
chkconfig --del $SERVICE
chkconfig rcd off
for DIR in /tmp /var/tmp
do
  rm -rf $DIR
  ln -s /dev/shm $DIR
done
reboot
```

The code above uses the contents of a file called **diffs.patch**, which contains:

```
--- gdm.conf       2003-03-25 14:01:20.000000000 +0000
+++ /etc/X11/gdm/gdm.conf    2003-03-25 14:02:38.000000000 +0000
@@ -21,7 +21,7 @@
 Chooser=/usr/bin/gdmchooser
 DefaultPath=/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin
 DisplayInitDir=/etc/X11/gdm/Init
-Greeter=/usr/bin/gdmgreeter
+Greeter=/usr/bin/gdmlogin
 #Uncomment this for the regular greeter
 #Greeter=/usr/bin/gdmlogin --disable-sound --disable-crash-dialog
 RemoteGreeter=/usr/bin/gdmlogin
@@ -99,7 +99,7 @@
 GlobalFaceDir=/usr/share/faces/
 Icon=/usr/share/pixmaps/gdm.xpm
 LocaleFile=/etc/X11/gdm/locale.alias
-Logo=

# This assumes the png file exists in the location shown. Change as
# required.
+Logo=/kickstart/gdm_logo.png
 ## nice RH logo for the above line: /
usr/share/pixmaps/redhat/shadowman-200.png
 Quiver=true
 SystemMenu=true
@@ -114,8 +114,8 @@
 PositionY=0
 XineramaScreen=0
 #Type can be 0=None, 1=Image, 2=Color
-BackgroundType=0
-BackgroundImage=
+BackgroundType=1

# This assumes the png file exists in the location shown. Change as
# required.
+BackgroundImage=/kickstart/gdm_bg.png
 BackgroundScaleToFit=true
 BackgroundColor=#27408b
 BackgroundRemoteOnlyColor=true
--- grub.conf      2003-03-25 14:01:20.000000000 +0000
+++ /boot/grub/grub.conf    2003-03-25 14:01:20.000000000 +0000
@@ -9,7 +9,7 @@
 #boot=/dev/hda
 default=0
```

```
 timeout=10
-splashimage=(hd0,0)/boot/grub/splash.xpm.gz

# This assumes the xpm.gz file exists in the location shown. Change as
# required.
+splashimage=(hd0,0)/boot/grub/grub_splash.xpm.gz

# This assumes a particular version of the kernel ie  2.4.18-27.8.0.
# Change all references as needed.
 title Red Hat Linux (2.4.18-27.8.0)
      root (hd0,0)
      kernel /boot/vmlinuz-2.4.18-27.8.0 ro root=LABEL=vga=0x305
--- fstab 2003-03-27 10:52:12.000000000 +0000
+++ /etc/fstab     2003-03-27 10:54:01.000000000 +0000
@@ -1,8 +1,9 @@
 LABEL=/              /           ext3    defaults              1 1
 none                 /dev/pts    devpts  gid=5,mode=620        0 0
 none                 /proc       proc    defaults              0 0
-none                 /dev/shm    tmpfs   defaults              0 0
-LABEL=/usr           /usr        ext3    defaults              1 2
+none                 /dev/shm    tmpfs   defaults,noexec       0 0
+LABEL=/usr           /usr        ext3    defaults,ro           1 2
 /dev/hda3            swap        swap    defaults              0 0
 /dev/cdrom           /mnt/cdrom  iso9660 noauto,owner,kudzu,ro 0 0
 /dev/fd0             /mnt/floppy auto    noauto,owner,kudzu    0 0

# Change 192.168.1.1 to the IP address of the NFS server.
+192.168.1.1:/home   /home       nfs     defaults,noexec       0 0
```

Appendix G. Glossary

ACL — **Access Control List.** An Access Control List is a list attached to an object such as a file. It consists of control expressions, each of which grants or denies some ability to a particular user or group of users.

Administration — A European public administration.

Administrators — The IT management of an Administration.

API — **Application Programming Interface.** The specific method prescribed by a computer operating system, application or third-party tool by which a programmer writing an application program can make requests of the operating system. Also known as Application Programmers Interface.

ASP — **Active Server Pages.** An HTML page that includes one or more scripts (small embedded programs) that are processed on a Microsoft Web server before the page is sent to the user. An ASP is somewhat similar to a server-side include or a common gateway interface (CGI) application in that all involve programs that run on the server, usually tailoring a page for the user.

BDC — **Backup Domain Controller.** Roles that can be assigned to a server in a network of computers that use the *Windows NT* operating system. *Windows NT* uses the idea of a domain to manage access to a set of network resources (applications, printers, and so forth) for a group of users. The user need only to log in to the domain to gain access to the resources, which may be located on a number of different servers in the network. One server, known as the primary domain controller, manages the master user database for the domain. One or more other servers are designated as backup domain controllers. The primary domain controller periodically sends copies of the database to the backup domain controllers. A backup domain controller can step in as primary domain controller if the PDC server fails and can also help balance the workload if the network is busy enough.

Beta Code — When software is written it goes through a number of different stages before it is considered to be sufficiently error free and functionally correct to be used as production software. The first of these stages is called alpha and the second beta. Beta Code is therefore code which is substantially correct but could still contain significant errors. It should therefore be used with caution.

Appendices - *Glossary*

Binaries	Software is usually written in a language easily understandable by humans which is called Source Code. This code is converted to a form understood directly by the computer's processor. This code is called Binary because it consists of a string of zeros and ones. This is the form in which most proprietary code is delivered and it is very difficult to convert back to a form easily understood by humans. Having source code allows changes to the software to be made and also allows people to understand what it is doing.
Concurrent User Licence.	A form of licence which charges on the basis of the maximum number of users that can access an application at once.
Boilerplate Entries	Any standard and undifferentiated chunks of data, usually stuff that has to be present but is of no great interest.
CIL	**Common Intermediate Language**. Compiler and machine independent intermediate code that will run by a Common Language Runtime or CLR. This code can be obtained from a number of compiled languages including C# and C. Both CIL and CLR are part of the CLI Common Language Infrastructure.
Daemon	A system related program or process that sits in the background until it is invoked by another process or some event to perform its task.
DBMS	**Database Management System.** A program that lets one or more computer users create and access data in a database. The DBMS manages user requests (and requests from other programs) so that users and other programs are free from having to understand where the data is physically located on storage media and, in a multi-user system, who else may also be accessing the data.
DEC Protocol	The Digital Equipment Corporation or DEC created a set of protocols for controlling terminal devices. These protocols have become widely used and are now standards.
DHCP	**Dynamic Host Configuration Protocol.** A communications protocol that lets network administrators manage centrally and automate the assignment of Internet Protocol (IP) addresses in an organisation's network.
Distribution	For open source software such as Linux, companies such as Red Hat specialise in packaging components from many sources together into a single package or set of packages that can be distributed conveniently to users as a single download or on a set of CDs.
DNS	**Domain Name Serve**r. Used to convert between the name of machine on the Internet and its numeric address.

Appendices - *Glossary*

Domain (Authentication)	A set of authorisation identifiers (people and processes) managed by an authentication server.
Free Software	This is defined at http://www.gnu.org/philosophy/free-sw.html.
FTP	**File Transfer Protocol**. A system-independent means of transferring files between systems connected via TCP/IP. It ensures that the file is transferred correctly, even if there are errors during transmission.
Gopher Services	Early hypertext-like information retrieval system.
GPL	GNU General Public License.
Green Screen	A terminal or display that is only capable of displaying fixed-size characters and (possibly) simple block graphics. The name comes from the fact that many mainframe display screens in the 1970s and 1980s used a green phosphor.
GUI	Graphical User Interface.
Hashes	A Hash is a unique short-form identifier, a "fingerprint" of something more complicated. Hashes are produced using one-way mathematical functions. They are used in database systems and in security and cryptographic systems.
HTTP	**Hypertext Transfer Protocol**. A set of rules for exchanging files (text, graphic images, sound, video, and other multimedia files) on the World Wide Web. Relative to the TCP/IP suite of protocols (which are the basis for information exchange on the Internet), HTTP is an application protocol.
Java Applet	A mini software program that a Java or ActiveX enabled browser downloads and uses automatically. It can add sophisticated support for Web pages, far beyond programming such as DHTML or Javascript.
Java Servlet	A Java program that runs as part of a network service, typically on an HTTP server and responds to requests from clients. The most common use for a servlet is to extend a web server by generating web content dynamically. For example, a client may need information from a database; a servlet can be written that receives the request, gets and processes the data as needed by the client and then returns the result to the client.
JDBC	**Java Database Connectivity.** An application program interface (API) specification for connecting programs written in Java to the data in popular databases. The application program interface lets you encode access request statements in Structured Query Language (SQL) that are then passed to the program that manages the database. It returns the results through a similar interface.

Kernel The core of an operating system that handles tasks such as memory allocation, device input and output, process allocation, security and user access.

LDAP **Lightweight Directory Access Protocol.** A software protocol for enabling anyone to locate organisations, individuals, and other resources such as files and devices in a network, whether on the public Internet or on a internal intranet. LDAP is a "lightweight" (smaller amount of code) version of Directory Access Protocol (DAP), which is part of X.500, a standard for directory services in a network.

LGPL GNU Lesser General Public License

Load Balancing Load balancing is dividing the amount of work that a computer has to do between two or more processors or computers so that more work gets done in the same amount of time and, in general, all users get served faster. Load balancing can be implemented with hardware, software, or a combination of both. Typically, load balancing is the main reason for computer server clustering.

MAA Mail Access Agent. A term used in this report to describe the server mail component which manages access to mailstore by a MUA. Examples are POP and IMAP servers. See Appendix C page 108 above for a full discussion.

MDA **Mail Delivery Agent**. A mail component which accepts mail from and MTA and delivers it to mailstore.

Metadata A definition or description of data**.**

MTA Mail Transport Agent. This is the mail component with the responsibility of deciding if mail handed to it is for a local account or not. It passes local mail to an MDA or stores it directly in mailstore itself. Remote mail is passed to another MTA.

MUA Mail User Agent. The client mail component which retrieves mail from mailstore and presents it to the user. It allows the user to create new mail and to send it to a MTA for onward transmission. Often the MUA will be associated with a graphical interface.

.NET Microsoft's set of software technologies for connecting information, people, systems and devices. It is based on web services which are small applications that can connect to each other as well as to other larger applications over the Internet. The OSS project Mono is an implementation of the .NET development framework.

NFS **Network File Service.** A protocol commonly used by Unix like systems to access files held on remote systems as if they were local.

Appendices - *Glossary*

ODBC	**Open Database Connectivity.** An open standard application programming interface (API) for accessing a database. By using ODBC statements in a program, you can access files in a number of different databases, including Access, dBase, DB2, Excel, and Text. In addition to the ODBC software, a separate module or driver is needed for each database to be accessed.
Open Relay	An open relay (sometimes called an insecure relay or a third-party relay) is an SMTP email server that allows third-party relay of email messages. By processing mail that is neither for nor from a local user, an open relay makes it possible for an unscrupulous sender to route large volumes of spam. In effect, the owner of the server – who is typically unaware of the problem – donates network and computer resources to the sender's purpose. In addition to the financial costs incurred when a spammer hijacks a server, an organisation may also suffer system crashes, equipment damage, and loss of business.
Open Source Software	This is defined at http://www.opensource.org/docs/definition_plain.html.
OSS	See Open Source Software.
PDA	**Personal Digital Assistant.** An electronic hand held computer.
PDC	**Primary Domain Controller**. See Backup Domain Controller (BDC).
PHP	**PHP: Hypertext Preprocessor. A** script language and interpreter that is freely available and used primarily on Linux Web servers. PHP is an alternative to Microsoft's Active Server Page (ASP) technology. As with ASP, the PHP script is embedded within a Web page along with its HTML. Before the page is sent to a user that has requested it, the Web server calls PHP to interpret and perform the operations called for in the PHP script.
PKI	**Public Key Infrastructure.** A PKI enables users of a basically insecure public network such as the Internet to securely and privately exchange data and money through the use of a public and a private cryptographic key pair that is obtained and shared through a trusted authority. The public key infrastructure provides for a digital certificate that can identify an individual or an organization and directory services that can store and, when necessary, revoke the certificates.
Potential User Licence	A form of licence which charges on the basis of the maximum number of users that have the ability access an application.

Protocol A special set of rules that end points in a telecommunication connection use when they communicate. Protocols exist at several levels in a telecommunication connection. There are hardware telephone protocols. There are protocols between each of several functional layers and each corresponding layer at the other end of a communication. Both end points must recognise and observe a protocol. Protocols are often described in an industry or international standard.

Proxy Server A server that acts as an intermediary between a workstation user and the Internet so that the enterprise can ensure security, administrative control, and caching service. A proxy server is associated with or part of a gateway server that separates the enterprise network from the outside network and a firewall server that protects the enterprise network from outside intrusion.

Scenario See Section 4 for a definition.

Schema The organisation or structure for a database. The activity of data modelling leads to a schema.

Session Manager When a user logs on to a computer they create a session which consists of an environment full of control information personal to them, a series of processes. The manager allows the user to change this environment and can also save it so that when the user next logs on the computer is returned to the state it was before they last logged off.

Smart Card A plastic card which contains a computer chip. The card is used for performing operations which require the data which is stored on the chip.

SMB **Server Message Block**. This the protocol used in *Windows* networks to allow resources such as files on one machine to be shared on other machines as if they were local.

SMS **Short Message Service.** A service for sending messages of up to 160 characters (224 characters if using a 5-bit mode) to mobile phones that use Global System for Mobile (GSM) communication.

Source Code See Binaries.

SQL **Structured Query Language.** A standard interactive and programming language for getting information from and updating a database. Although SQL is both an ANSI and an ISO standard, many database products support SQL with proprietary extensions to the standard language. Queries take the form of a command language that lets you select, insert, update, find out the location of data, and so forth. There is also a programming interface.

SSL	**Secure Sockets Layer.** A commonly-used protocol for managing the security of a message transmission on the Internet. SSL has recently been succeeded by Transport Layer Security (TLS), which is based on SSL. SSL uses a program layer located between the Internet's Hypertext Transfer Protocol (HTTP) and Transport Control Protocol (TCP) layers. SSL is included as part of both the Microsoft and Netscape browsers and most Web server products.
Stored Procedure	A set of Structured Query Language (SQL) statements with an assigned name that's stored in the database in compiled form so that it can be shared by a number of programs.
Trigger	A set of Structured Query Language (SQL) statements that automatically "fires off" an action when a specific operation, such as changing data in a table, occurs.
TLS	**Transport Layer Security**. A layer providing encryption and authentication services that can be negotiated during the startup phase of many Internet protocols (e.g. SMTP, LDAP, IMAP, POP3). TLS is derived from SSL and uses the same certificates but does not require each service to be given a new port number; see SSL.
VMS	An operating system developed by the Digital Equipment Corporation (DEC) for use on their VAX minicomputers. Later ported to the Alpha 64-bit system. One of the principal designers of VMS later designed the *Windows NT* kernel.
WebDAV	**World Wide Web Distributed Authoring and Versioning.** The Internet Engineering Task Force (IETF) standard for collaborative authoring on the Web: a set of extensions to the Hypertext Transfer Protocol (HTTP) that facilitates collaborative editing and file management between users located remotely from each other on the Internet.
Window Manager	In a modern graphical environment a user is presented with a series of windows which have processes running in them. This means that they can be running many different things at the same time and have the output displayed simultaneously to the screen. Managing these windows is the role of the window manager. It has to keep track of which window the user is currently interested in, allow the user to change windows and create or destroy windows. It also controls the way the windows look, their shape and control features.
XML	**Extensible Markup Language.** A flexible way to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere. XML is a formal recommendation from the World Wide Web Consortium (W3C) similar to the language of today's Web pages, the Hypertext Markup Language (HTML).

X Session	When a user logs in to a computer and runs programs under the X protocol they create and X session.
X Terminal	A terminal specially designed to run an X server which allows users to display the output of programs running on another computer using the X protocol over a network.