UNIX Code Migration Guide

UNIX Application Migration Guide

patterns & practices
proven practices for predictable results

Chapter 1: Introduction

Larry Twork, Larry Mead, Bill Howison, JD Hicks, Lew Brodnax, Jim McMicking, Raju Sakthivel, David Holder, Jon Collins, Bill Loeffler
Microsoft Corporation

October 2002

Applies to:
Microsoft® Windows®
UNIX applications

The *patterns & practices* team has decided to archive this content to allow us to streamline our latest content offerings on our main site and keep it focused on the newest, most relevant content. However, we will continue to make this content available because it is still of interest to some of our users.
We offer this content as-is, without warranty that it is still technically accurate as some of the material is undoubtedly outdated. Note that the content may contain URLs that were valid when originally published, but now link to sites or pages that no longer exist.

Summary: Chapter 1: Introduction is an introduction to the UNIX Migration Guide. It describes the intended audience and outlines the remainder of the document flow and organization. (13 printed pages)

Contents

Introduction
Contents of the Guide
Document Conventions
Background
Why Migrate?
Migration and Coexistence Options
Windows 2000 Benefits
Summary

Introduction

Welcome to the UNIX Migration Guide. This guide is designed to give you the best information available about the issues that you are likely to face if you are contemplating or have decided on greater integration between your UNIX and Microsoft® Windows® operating system environments. If you are a senior IT decision maker, network manager or operating system administrator, then this guide has been designed with you in mind. In addition, a large part of this document covers the detailed changes that need to be made at a coding level, making it a valuable resource for developers.

In the first few chapters, you will look at the planning and practical issues involved in migration or co-existence between UNIX and Windows. These chapters discuss whether a move to Windows is the best choice for your organization and provide a review of the different ways in which such a migration can be done.

Chapters 6 through 15 have sections for both UNIX and Windows programmers. If you are a UNIX programmer, you will see how you can adapt your code so that it can be recompiled to run in a Windows Win 32 or Native UNIX Interix environment. If you are a Windows programmer, there are chapters that cover how you can port UNIX functions to Windows and how you can ease the process of rewriting an application within the Microsoft Win32® API framework.

This guide provides information that is both practical and prescriptive. If you need more in-depth discussions of the concepts behind this material, you should refer to resources such as the Microsoft .NET Framework Software Development Kit (SDK), the .NET document available in the Microsoft Visual Studio® development system and the Windows 2000 Resource Kit.

This guide includes information from consultants working in the field and from organizations that have already confronted and solved the issues around migrating from UNIX to Windows to produce what is currently defined as best practice. We hope you enjoy reading this guide and that you find the material helpful, informative and interesting.

Contents of the Guide

This guide consists of the following 15 chapters:

- **Chapter 1—Introduction**. This chapter sets the scene for the remainder of the guide, defining the contents and creating the framework for the remaining chapters. It introduces the concepts behind UNIX migration and coexistence and presents the ideas that will be discussed in greater detail in the body of the guide. You will look at the reasons why you should consider moving from UNIX to Windows. In addition, the chapter provides case histories that may be similar to your own environment, along with a description of the migration approach that each organization took.
- **Chapter 2—UNIX and Windows Compared**. This chapter covers the technical background behind the evolution of UNIX and Windows, pointing out the differences and similarities between the two environments. Chapter 2 also looks at a number of background technologies that can form part of your migration or coexistence plans.
- **Chapter 3—The Migration Process**. This chapter takes a detailed look at the migration process and works through the structure that any migration project requires. This will permit you to produce a detailed vision of the effect that migration will have on your organization. Finally, you will be able to create an overall high-level vision document that will serve as the skeleton for the next two chapters.
- **Chapter 4—Assessment and Analysis**. You now need to look in detail at the application or applications that you want to move to Windows. Chapter 4 discusses the factors that indicate whether an application can coexist, can migrate or must be rewritten. It covers the components of the UNIX environment, their equivalents in Windows and how easy it is to convert one to the other. You can then insert this information into your migration plan structure.
- **Chapter 5—Planning the Migration**. This chapter allows you to put the final touches to your

plan, combining the high-level approach in Chapter 3 with the detailed appraisal of your application in Chapter 4. On completing this chapter, you will have the complete documented framework for your migration that will help to reduce risk and maximize the probability of a successful transition to the Windows environment.

- **Chapter 6—UNIX and Windows Interoperability**. This chapter focuses on enabling Windows and UNIX applications to work together. It provides a high-level overview of the considerations that you must address when you have decided that interoperability best suits the needs of your organization.
- **Chapter 7—Creating the Development Environment**. Aimed at both the IT decision maker and the developer, this chapter lists the various tools and technologies that are currently available to make your chosen approach work. If you are a decision maker, it will allow you to make the correct budgetary considerations. If you are a developer, it will give you a foundation in the tools that you will use to migrate, modify or rewrite your code.
- **Chapter 8—Preparing for Migration**. Pre-migration deals with a more detailed analysis of the steps that you need to take before you carry out the code conversion itself. Aimed at developers, this section covers such areas as source code standards compliance and script migration, and introduces a range of best practices that will ease later operations in the migration cycle.
- **Chapter 9—Win32 Code Conversion**. This extensive chapter works through the fundamentals of converting code from UNIX to Windows. Processes, threads, signal handing, memory management, networking and a host of related subjects are all illustrated with examples.
- **Chapter 10—Interix Code Conversion**. Interix provides an environment that allows a large number of UNIX applications to run on Windows with little or no adjustment. Chapter 10 deals with the changes that you may need to make so that your current applications can run on Interix. This approach allows interoperation between Windows and UNIX applications, without the additional effort of a full code port.
- **Chapter 11—Migrating the User Interface**. For end users, the user interface is the main concern when using the migrated application. This chapter covers the elements that make up good interface design and tells you how to implement UNIX graphical user interface elements in Windows. A significant section deals with the X-Windows environment and the particular challenges that arise when migrating from X-Windows to Windows.
- **Chapter 12—Testing and Quality Assurance**. Before you can release your newly migrated application, you must ensure that it meets your quality standards. Chapter 12 deals with the creation of a test environment, designing the test plan and evaluating the results. It then covers the release process and release criteria that verify the application's quality and suitability for a production environment.
- **Chapter 13—Creating the Live Environment**. In Creating the Live Environment, you look at methods for deploying your applications efficiently, ranging from the Windows Installer service to software policies to Systems Management Server. You will also cover additional areas such as maintenance, support and user training, all of which make a vital but often overlooked contribution to the success of a migration plan.
- **Chapter 14—Migrating Fortran**. Migrating Fortran code from a UNIX environment to Windows or to Interix brings a slightly different set of challenges. This chapter looks at the Fortran migration options and factors that you will need to consider when making such a move.
- **Chapter 15—Roadmap for Future Migrations**. This final chapter examines the future for application development using Microsoft technologies. The introduction of the .NET Framework has significantly streamlined the production of high-quality, secure code and gives developers far greater freedom to develop in any language while making use of common services.

Document Conventions

This guide uses the style conventions and terminology shown in Table 1.

Table 1. Document conventions

Element	Meaning
bold font	Characters that you type exactly as shown, including commands and switches. Programming elements, such as methods, functions, data types and data structures appear in bold font (except when part of a code sample, in which case they appear in monospace font). User interface elements are also bold.
Italic font	Variables for which you supply a specific value. For example, *Filename.ext* could refer to any valid file name for the case in question. New terminology also appears in italic on first use.
`Monospace font`	Code samples.
%SystemRoot%	The folder in which Windows 2000 is installed.
Tip	Alerts you to supplementary information that is not essential to the completion of the task at hand.
Note	Alerts you to supplementary information.
Important	Alerts you to supplementary information that is essential to the completion of a task.
Caution	Alerts you to possible data loss, breaches of security or other more serious problems.
Warning	Alerts you that failure to take or avoid a specific action might result in physical harm to you or to the hardware.

Background

Since the mid 1980s, corporate data centers have been moving away from mainframes running dedicated operating systems to mini-computers, often using one or other of the myriad flavors of UNIX. At the same time, the users' experience of these systems has, in many cases, stayed the same, involving text-based interaction with dumb terminals or a terminal-emulation session on a PC.

More recently, IT managers have questioned this approach and have been looking at changes in the UNIX marketplace and the increasing expense of being tied in to single-vendor software and hardware solutions. The growth of Linux as a lightweight version of UNIX has added fuel to this interest, raising the number of organizations who are considering a migration to alternative platforms.

While there may seem to be overwhelming reasons for organizations to consider migrating to Linux as the most obvious path, there are significant issues in taking this approach. When analyzed in depth, migration or coexistence with the Microsoft Windows operating system can make more sense in the long run, offering greater enterprise readiness and a more highly integrated future roadmap for application development and server support, along with quantifiable reductions in total cost of ownership (TCO).

Audience

The intended audience for this guide consists of two groups: firstly, Corporate Information Officers

(CIOs), Information Technology (IT) directors, data center managers and senior engineers assessing an organization's UNIX migration options and, secondly, the developers tasked with making such a migration a reality.

If you fall into the decision-maker camp, then chapters 2 to 7 cover your area of primary focus. If you are a developer, then chapters 9 to 14 contain the detailed information that you need. Finally, chapters 7 and 15 include content of interest to both audiences.

Chapter Start Point

The start point for this chapter is that you are an IT decision-maker or developer either assessing or involved in a UNIX co-existence or migration project.

Chapter End Point

By the end of this chapter, you should understand the concepts behind UNIX-to-Windows migration that will be discussed in greater detail throughout this guide. You will be aware of the different ways in which you can integrate UNIX and Windows, from a simple recompile to extended coexistence. You will have been introduced to some of the tools that enable this co-operation and have an understanding of the scope of the remainder of the guide.

System Prerequisites

As this guide is aimed at two separate audiences, the material itself has differing prerequisites. Chapters 2 to 7 assume a strong familiarity with planning and executing major infrastructure projects that intrinsically carry an enhanced level of risk.

Chapters 9 to 13 assume a detailed knowledge of either the UNIX or the Win32 development environments. This includes familiarity with the C, C++ or Fortran languages and syntax and the ability to write new code or to analyze and adapt existing code to fit the migration or co-existence scenario.

Out-of-Scope Topics

Because migration to UNIX can encompass a vast range of topics, this guide deals with the core areas of code ports and migrating applications to the Win32 environment. Therefore, this guide does not deal with the following topics:

- Administrative migrations (for example, moving user accounts)
- Database migrations, such as Oracle or MySQL to Microsoft SQL Server™ 2000
- Migrating Java 2 Enterprise Edition to Visual Studio or .NET
- Linux-Apache-MySQL-PHP or iPlanet to Windows
- Infrastructure concerns
- Applications written in languages other than C, C++ or Fortran
- Apache to Microsoft Internet Information Server (IIS) migration and converting Common Gateway Interface (CGI) programs to ISAPI.NET

For more information on these topics, you should refer to the following Web sites:

- Migrating Oracle Databases to SQL Server 2000
- Microsoft Hotmail Migration Techinical Case Study

Why Migrate?

In the 50-year history of the IT industry, there has been only one overriding constant: the continual presence of change, both in terms of the technology and the capability of computer systems. Few other areas in business have experienced such rapid development and often terrifying levels of obsolescence, with equipment that was purchased new at $10,000 barely able to recoup $200 a mere three years later—a 98% depreciation rate.

The result of this continuing progress is that you as the IT decision maker are caught in a difficult situation. You can make no changes and risk your systems slipping into obsolescence, or you can make a change and risk joining a computing trend that turns out to be an evolutionary dead end.

The Changing Situation

The various implementations of the UNIX operating system have served industry well, as witnessed by the very large base both of installed systems and large-scale applications installed on those systems. However, there are increasing signs of dissatisfaction with expensive, often proprietary solutions and a growing sense that perhaps the concept of "big iron" has had its day—in the same way as it has for most of the mainframes of the type portrayed in 1970s science fiction films.

One of the most extraordinary and unexpected successes of the Intel PC architecture is the extent to which this basic framework has been extended to encompass very large server and data center environments. Large-scale hosting companies are now offering enterprise-level services to multiple client companies at availability levels of over 99.99 percent on what are simply racks of relatively cheap PCs. Technologies such as clustering, Network Load Balancing (NLB) and Component Load Balancing (CLB) enable the humble personal computer to take on and match the levels of throughput, availability and reliability of all but the most expensive "big iron" solutions and the supercomputers.

Why Move from UNIX?

So what reasons are driving you to consider moving from your cozy UNIX environment that has served you so faithfully all these years? Three main reasons spring to mind:

- Reduced costs
- Increased flexibility
- Improved performance

Let's examine each of these areas briefly:

Reduced costs

For most organizations, IT is a business enabler and a means to greater productivity, rather than a revenue-earner. For this reason, IT services are a cost to the business and, like any other cost, need to deliver the highest possible levels of business value.

When assessing IT costs, you are likely to be looking beyond just the initial cost of acquisition to the total costs over a 5, 10 or 15 year lifecycle, incorporating support and maintenance costs into the equation. Therefore, when considering costs, you are likely to be looking for a solution that provides lower total cost of ownership, rather than the fact that, say, the operating system is free.

Increased flexibility

In the current shrinking economy, businesses have to be more flexible and able to react faster to remain competitive. Increasingly, it is the organization that can turn more sharply and realign itself to its clients' needs fastest that wins the business. If you can provide the best infrastructure that lets your engineers, analysts or scientists work to their fullest potential, then you are giving your organization the competitive edge that it needs to survive and thrive.

Greater performance

Increasingly, applications such as computer aided engineering, risk analysis and 3-D modeling and rendering are becoming mainstream tools, thus putting greater and greater demands on your computing infrastructure. You need to ensure that your equipment can scale effectively and provide the price-to-performance ratios so that you can give your organization's employees the horsepower that they need to carry out their jobs efficiently.

What Are the Options?

The fact is that whatever your current environment, you cannot afford to avoid considering what you should be doing in the future. This is necessarily a continuous process, as technology and business situations change. So, once you have made the decision that you need to investigate migration as a possible business strategy, what are your options?

In the current market, the alternatives are either to migrate to Linux or to Windows. Let's look briefly at each option.

Linux

For organizations with a large installed base of UNIX applications, moving to Linux would initially seem to be attractive. The immediate benefits would be that you could:

- Migrate UNIX applications with minimal changes
- Move to PC-based architecture, thus reducing hardware costs
- Acquire an operating system at little or no apparent cost

While it cannot be denied that migrating your applications to Linux with minimal changes is a compelling reason to consider this route, it is worth considering whether the disadvantages outweigh the benefits.

There are a number of companies that provide their own distribution of Linux, so you have almost as much choice of Linux builds as variants of UNIX. Linux certainly is making great strides in the marketplace, primarily as the operating system for smaller Web sites where a free OS is a major factor, but does that make Linux a suitable choice for your enterprise?

To make a true assessment of the suitability of Linux, you need to look at the following areas.

- Do I need an enterprise-wide directory service?
- Do I need to support clustering or load-balancing?
- Will I need to integrate with a heterogeneous environment?
- Will I need to use features on Linux that will tie me in to a single vendor?

- Do I need consistent, integrated, enterprise management?
- Will I require a well-defined enterprise roadmap of future innovation and features?

If you answer yes to any of these questions, then you may find that Linux provides a less than ideal solution.

Windows

For many from a UNIX background, just thinking of migrating to Windows is tantamount to treason. However, if you take a look at the complete picture, you may find that there are significant advantages in taking this path. The main advantages of taking the Windows route are that you receive:

- The best price-to-performance ratio
- The lowest TCO
- An enterprise-level directory service
- Integrated management and security model
- Rapid application development tools
- Built-in clustering and high availability technologies
- Worldwide enterprise support
- Large network of trained consultants

While this list represents just a fraction of the potential benefits of moving to the Windows environment, these are probably some of the factors highest on your list. However, there is one main area of concern that is not covered here, that of migrating existing UNIX code to the Windows platform, which is covered in the following section on migration and coexistence options.

For more information, see the white paper The Benefits of Migrating from UNIX to Windows 2000.

Evaluating the Long Term Risk

Ultimately, the decision on which path to take will be highly dependent on the risks associated with each course of action.

Specifically, do you:

- Go for easier migration to a more uncertain operating environment?

  —or—

- Look beyond the immediate migration task to the longer-term support and stability of your chosen infrastructure?

Understanding the Linux revenue model

There are many companies starting to compete on providing Linux-based solutions and the Linux builds that they distribute are becoming increasingly proprietary in nature. As these businesses provide the operating system for nothing or at a nominal fee, they must necessarily base their revenue models heavily on providing support rather than on the supply of the software itself.

The economic realities of this business model mean that either you pay them for support and they flourish, or you solve your problems in-house and your supplier quietly exits. While this guide does not pretend to be an impartial reference, this is an argument that needs serious consideration. Taking this argument to its logical conclusion, the most successful Linux distributor of the future is likely to be the one that gets paid the most for support.

The Microsoft revenue model

By charging for licensing its applications and operating systems, Microsoft is not dependent on support revenues for financial stability. Hence Microsoft expects to be a constant in the future IT landscape, continuing to work with hardware and third-party software vendors to enhance and develop both the .NET Framework and the forthcoming .NET Server family.

Migration and Coexistence Options

With a large code base of installed UNIX applications, you are probably unlikely to relish the thought of throwing the entire environment out and starting again with an unfamiliar platform. Fortunately, as this guide shows, you do not necessarily have to, as there are methods that will allow you to preserve your investment in UNIX applications while developing under or moving to Windows over a longer period.

This guide deals with the four possible options for migration or co-existence between UNIX and Windows. These are:

- Quick port
- Full port
- Application rewrite
- Coexistence

Quick Port

One of the simplest migration paths possible is to port the code directly to Microsoft Services for UNIX 3.0. Services for UNIX 3.0 includes Microsoft Interix, which provides a UNIX environment that runs on top of the Windows kernel, allowing your native UNIX applications and scripts to work alongside Windows applications.

Full Port

The full port involves migrating your software to Windows with the minimum changes necessary to allow the application to run. Unlike the quick port, this does entail alterations to the source code, although the number of modifications will depend on the level of standards compliance within the original application.

Application Rewrite

Rewriting the application is the ideal approach if you want to make full use of all the benefits of migrating to the Windows platform. While initially requiring the greatest amount of work, this course also promises the greatest rewards.

Coexistence

With coexistence, you retain the original application alongside the new version while simultaneously porting or rewriting the application. Taking this approach significantly reduces risk, as it allows you to use the original system should unexpected issues appear with the new application. However, you will need to employ a cross-platform source code control system to allow concurrent development on both UNIX and Windows simultaneously.

Chapter 3, The Migration Process, and Chapter 4, Assessment and Analysis, cover these options in greater detail, allowing you to decide which is the most appropriate choice or combination of choices for your particular environment.

Windows 2000 Benefits

Windows 2000 is Microsoft's most scalable enterprise operating system yet released and is currently the best platform for hosting applications migrated from a UNIX environment. A significant advance over Windows NT 4.0, Windows 2000 provides a stable, scaleable and reliable operating system that can run the largest corporate data centers.

Scalability

Windows 2000 encompasses both horizontal and vertical scalability, meaning it can scale both in terms of the number of users that it can contain within its directory and in the number of simultaneous connections it can host. With Windows 2000, you can both scale up (by adding more processors or faster hardware) and scale out (by adding multiple computers to increase overall capacity and using clustering technologies where appropriate). This dual scalability allows Windows 2000 to match or exceed UNIX performance levels at significantly lower costs.

Directory size

Microsoft Active Directory® directory service removes the 40,000 user account per domain restrictions of Microsoft Windows NT® 4.0 and gives you the ability to scale to deployments well in excess of 1,000,000 user accounts. Laboratory testing of Active Directory has demonstrated the capacity to cope with a single forest incorporating up to 10,000,000 objects. Hence Windows 2000 can cope even with the user accounts of the largest organizations.

Multiple domain controllers

Active Directory provides logon scalability by letting you add more domain controllers. Multi-master replication keeps all the domain controllers synchronized, ensuring that all servers within a domain maintain an up-to-date copy of the directory database. Be your organization centralized or distributed, Windows 2000 can provide you with fast and secure logon facilities.

Reliability

Windows 2000 provides several features that enhance reliability, making it suitable for the hosting environment or mission critical, non-stop computing.

Domain controllers

Multi-master replication also means that all servers within a domain maintain a full writeable copy of the directory database. This, in turn, means that there is no dependency on a primary domain controller and no requirement for manual promotion of a backup domain controller should the primary fail.

> **Note** There are still operations masters such as the Schema master that need to be given a higher level of tender loving care. However, if an operations master does fail, there are well-documented procedures for seizing control of an operations master function.

Clustering services

Windows 2000 Advanced Server and Datacenter Server let you create either two-node (Advanced Server) or up to four-node (Datacenter Server) clusters. Clustering technology allows you to reboot nodes in the cluster while still providing services to the network.

Network load balancing

Windows 2000 NLB provides enhanced reliability, typically in the web and business logic tiers. Should a front-end server fail, the user's session will automatically route to another available computer. For ultimate reliability, you can use multiple NLB clusters, then load-balance client requests across the two clusters simply using DNS round robin.

Component load balancing

CLB is a feature of Microsoft Application Center 2000 and enables you to create load-balancing clusters within your business logic tier.

Security

Windows 2000 is designed with security as a main goal. From the isolation of the logon sequence from other processes to the granularity of control that can be applied via NTFS and Active Directory permissions and group policy, Windows 2000 provides a firm framework for implementing hosted applications.

Additionally, Windows 2000 security templates allow you to apply security settings easily in order to set different levels of protection to your domain controllers or member servers.

Administration

Windows 2000 has a number of administrative interfaces that enable administrators and developers to monitor and manage the operating system. Windows Management Instrumentation (WMI) provides a series of event monitors that can be used to provide real-time information on the health of a Windows 2000 computer. Microsoft Management Console (MMC) provides a framework for customizable tools that you can use to carry out a range of administrative tasks.

Summary

In this chapter, you were introduced to the contents of the remainder of the guide. You looked at the main factors behind considering a move from your current UNIX environment and the twin options of migrating to Linux or to Windows. You reviewed the advantages and disadvantages of each approach and the genuine issues that migration to Linux entails. You were introduced to the four migration and co-existence paths and covered the benefits of Windows 2000 as an operating environment.

Now it's time to move on to reviewing the backgrounds to both Windows and UNIX and their similarities and differences.

patterns & practices
proven practices for predictable results

Send feedback to Microsoft