UNIX Custom Application Migration Guide

Version 2.0

Volume 5: Deploy and Operate

Published: May 2006

Microsoft

Contents

About This Volume

Introduction to Volume 5

The *UNIX Custom Application Migration Guide* (UCAMG) is designed to provide the best information available about the recommended strategy for migrating UNIX applications to the Microsoft® Windows® operating system environment, starting with the initial planning of the migration and ending with its final deployment. Volume 1: *Plan* discussed how to apply the Envisioning and Planning Phases of the Microsoft Solutions Framework (MSF) Process Model when conducting a UNIX-to-Windows migration project. The build volumes (Volume 2, Volume 3, and Volume 4) applied the next phases in the Process Model—the Developing Phase and the Stabilizing Phase—directing them for using specific migration technologies.

This volume, Volume 5: *Deploy and Operate*, describes the steps involved in deploying the migration and the post-deployment operations activities. The operations guidance offered in this chapter is organized according to Microsoft Operations Framework (MOF), which is a complementary framework to MSF. The following topics are discussed in this volume:

- Deploying Phase
- Operations

Intended Audience

This volume is designed for senior IT decision makers and managers, network managers, project managers, operating system administrators and customer IT staff, who are responsible for maintaining the application after its deployment.

Knowledge Prerequisites

It is assumed that you have already read the “About This Guide” document, which provides an overview of the entire five-volume guide and navigational guidance. References to other volumes, especially the build volumes (Volume 2, Volume 3, or Volume 4) of this guide, are made in this document.

If you are unfamiliar with MOF, you will need to review it to understand the operational guidance that is provided in Chapter 2, “Operate” of this volume.

Note Information about MOF is available at http://www.microsoft.com/mof.

In addition to the background reading, you should posses the following knowledge prerequisites before reading this volume:

- Basic knowledge of the UNIX and Windows environments.
- Hands-on experience on Windows environments.
- Deployment procedures on Windows.

Layout of the Guide: Volume 5

The following diagram depicts the layout of the guide and how the volumes of the guide correlate with the MSF/MOF process components. The white-shaded portion depicts the position of the current volume in the layout of the entire guide.

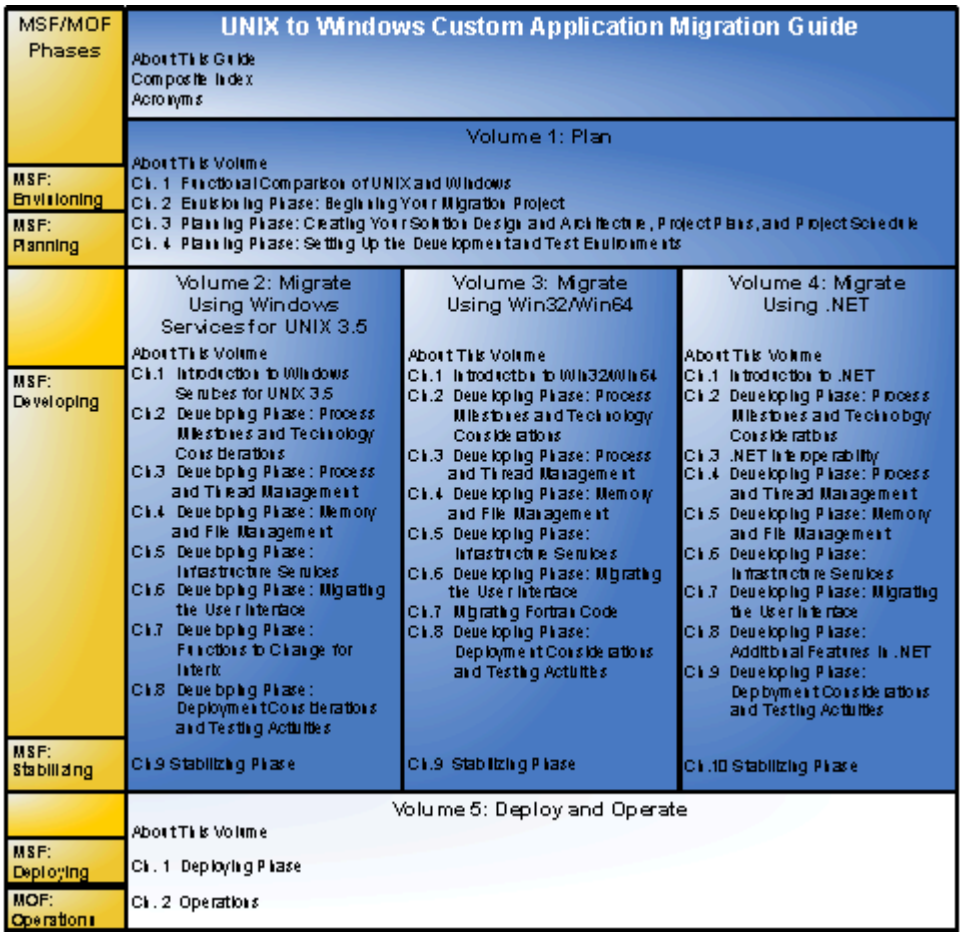

Figure 0.1. UCAMG organization

Organization of Content

- **About This Volume**. This chapter provides information on the organization of the volume and about its intended audience. It also lists the knowledge prerequisites required for this volume and provides resources, such as document conventions, used in this guide.
- **Chapter 1: Deploying Phase**. This chapter provides an overview of the activities that you need to perform during the MSF Deploying Phase including the major tasks, deliverables, and tools.
- **Chapter 2: Operations**. This chapter describes the post-deployment activities for maintaining the applications as per the MOF Process Model. It also provides a brief overview of further technologies and suggests further readings.

Resources

This section describes the various resources that are included in the *UNIX Custom Application Migration Guide* and information that will assist in using the guide.

Acronyms

Please see the Acronyms list accompanying this guide for a list of the acronyms and their meanings used in this volume.

Document Conventions

The document conventions used in this volume are primarily designed to help you to quickly identify the operating system and the interface (command line or graphical) being discussed. The platforms discussed in this volume are Microsoft Windows and UNIX. In general, Windows operating system commands are executed by clicking user interface (UI) elements, and these elements are visually distinguished in this volume by the use of bold text. In contrast, the UNIX operating system typically uses a command-line interface, and these instructions are visually distinguished in this volume by the use of monospace font.

These interface and execution differences are not absolute; and in cases where visual cues do not clearly delineate between operating systems, the text will clearly make this distinction.

Table 0.1 lists the document conventions used in this volume.

Table 0.1. Document Conventions

Text Element	Meaning
Bold text	Used in the context of paragraphs for commands; literal arguments to commands (including paths when they form part of the command); switches; and programming elements, such as methods, functions, data types, and data structures. Also used to identify the UI elements.
Italic text	Used in the context of paragraphs for variables to be replaced by the user. Also used to emphasize important information.
Monospace font	Used for excerpts from configuration files, code examples, and terminal sessions.
Monospace bold font	Used to represent commands or other text that the user types.
Monospace italic font	Used to represent variables that the reader supplies in command-line examples and terminal sessions.

Text Element	Meaning
Shell prompts	The MS-DOS® prompt is used in Windows.
Note	Represents a note.
`Code`	Represents code.

Chapter 1: Deploying Phase

Following the Stabilizing Phase, the team shifts its focus to the deployment activities required to place the solution into a production environment. The Deploying Phase culminates in the Deployment Complete Milestone, where the team obtains final customer approval of the project. This chapter discusses the various key activities of the Deploying Phase along with the major tasks, deliverables, and tools.

Goals for the Deploying Phase

The overarching goal of the Deploying Phase is to place the application into a production environment. Supporting goals include deploying the solution technology and components, stabilizing the deployment, and transferring the project to the operations and support teams. If the solution has been properly planned and developed before this phase begins, deployment is a routine activity. The groundwork for a smoothly functioning deployment was laid when the current application environment was assessed and documented during the Envisioning Phase and when the deployment plan was created during the Planning Phase. During the Deploying Phase, this deployment plan is further revised with the addition of more specific and up-to-date details about the deployment.

In some organizations, the project team is responsible for the entire deployment. In other organizations, there may be a group (such as a central engineering group or operations group) that is independent of the project team and who is responsible for deployment. For this reason, this chapter focuses on deployment tasks and issues instead of the persons responsible for these tasks. After deploying the application, the components are transferred from the test environment to the staging environment.

Major Tasks and Deliverables

Table 1.1 describes the major tasks that occur during the Deploying Phase and lists the owners responsible for achieving them.

Table 1.1. Deploying Phase Major Tasks and Owners

Major Tasks	Owners
Completing deployment preparations The team updates the deployment plan developed in the Planning Phase. The team also installs and configures the necessary software and hardware components for deployment.	Release Management and Development
Creating operations procedures The team creates documents and procedures to monitor and maintain the migrated application.	Release Management and Development
Deploying the solution The team deploys the core components of the application and enables access to the migrated application to all targeted users.	Release Management and Development
Stabilizing the deployment The team ensures that all the sites are operating and verifies with the customer that the entire solution is operating satisfactorily. This task also helps in tracking and resolving any issues.	Project team

Major Tasks	Owners
Transferring ownership to operations team The team formally transfers the responsibility of the migrated application to the operations team.	Release Management
Closing the Deploying Phase The team meets the Deployment Complete Milestone requirements and later completes post-project reviews with the customer and project team.	Project team

Completing Deployment Preparations

The project team can deploy the solution upon completion of the Stabilizing Phase. As the Stabilizing Phase nears completion, the Release Management lead assigns deployment tasks to the staff based on the deployment plan created during the Planning Phase.

The major activities involved with this task are:

- Creating the live environment.
- Deploying the migrated application.

You can typically deploy the Microsoft Windows environment using the Microsoft Windows Installer service. This method identifies a new deliverable in the form of an installer package (MSI) that can be applied by the Windows Installer service. Applications that target the Interix environment can rely on standard UNIX installation scripts. However, depending on the platform from which the application has been migrated, native binary programs for package management may be available.

The technologies for deploying Microsoft Interix, Microsoft Win32®, and Microsoft .NET applications are discussed in the build volumes (Volume 2, Volume 3, and Volume 4, respectively) of this guide. The following subsections only elaborate on the common activities shared by the three technologies: creating the live environment and deploying the migrated application.

Creating a Live Environment

This section focuses on the following topics related to implementing the migrated application:

- Interoperability
- Application deployment
- Network file systems and application servers

Interoperability

The team must address UNIX and Windows interoperability issues specific to the live environment. This is discussed in more detail in the “Operating a Mixed Environment” section in Chapter 2, “Operations” of this volume.

Application Deployment

For many organizations, deploying the new application is a major part of the migration project. Each of the build volumes (Volume 2, Volume 3, and Volume 4) of this guide provides a section, “Deployment Considerations,” in the “Deployment Considerations and Testing Activities” chapter, which discusses the tools and techniques that you can use to deploy the migrated application.

Networked File Systems and Application Servers

UNIX applications are often stored on an application server, and their executable files are accessed from the client using network file system (NFS) mounts. This greatly simplifies the

deployment of applications because they need to be deployed to only one server (the application server).

If your migrated application continues to use application servers, there are issues that affect both Win32/Win64 and Interix implementations. The main issues arise because of differences in the ways that the UNIX and Windows networked file systems operate.

Migration to the Windows platform provides an opportunity to reevaluate whether to continue using application servers where applications are retrieved from local execution. Your evaluation must take into account the network bandwidth requirements and the proximity of network shares. You must deploy any Win32-based application using the Windows Installer service, regardless of whether it is located on an application server. This may entail as little effort as placing a shortcut on the desktop of the user, although most applications require at least some skeletal installation to accommodate Component Object Model (COM) or Distributed Component Object Model (DCOM) component registration, user preference settings, and building local configuration files. Creating .msi installation packages enables an enterprise to maintain a consistent technology, use Group Policy-based deployment, and enjoy wide support by other vendors of other tools. Consider that most applications will enjoy better performance and reliability when executed from local drives.

The specific details pertaining to each Windows technology have been covered as part of the “Development Considerations for Deployment” section in the “Closing the Developing Phase” chapter in each of the build volumes.

Deploying the Migrated Application

The deployed application consists of the following components that need individual configuration and verification:

- **Physical environment**. During the Planning Phase, the project team determined how the solution architecture will be configured, the specific types of hardware and software that will be required, how the software will be installed and configured on each computer, and what other components will be required to deploy and maintain the solution. In the Deploying Phase, the actual new physical and service hardware is built, and the software is installed and tested.
- **System software**. The project team updates the documentation of all system software, such as operating system, database platform, application server, Web server, and networking protocols used for the migrated application.
- **Application software**. The project team installs the software applications required to support the environment and the migration application.
- **Custom application software**. The project team installs the custom application software created by the development team on the servers in the environment.

Tools for Deployment

This section gives you an overview about the tools for deployment and the installation instructions.

You will need to perform the following tasks to deploy the application and tools:

- Distribute the application.
- Install the application.

Depending on the scenario, these tasks can be carried out together, or you can select and implement just one of them at a time. The build volumes of the guide discuss specific tools for deployment.

Distribution of the Application

Distribution is the process of getting the binary of the application to the system on which it will be run. Distribution can be conducted by various methods, such as:

- Copying all the files required for the deployment using a script on the local system. This could be performed with a simple Windows script.
- Copying the script and binary or the package to the local system with specialized tools.

The first method is not efficient because it chokes up the network and does not provide reliability and scalability. Moreover, it is difficult to get a report of the errors. One way to overcome this issue is to use the second method: Copying the script and binary or the package to the local system with specialized tools such as Microsoft Active Directory® directory service or using such tools as Systems Management Server (SMS) for distribution.

Active Directory

Active Directory is part of the Windows Server™ 2003 operating system and is a core feature that not only handles security and privileges of the domain, but also has the capability to distribute MSI files. Group Policy settings can be set in Active Directory to distribute the required MSI file at any given time to the necessary computers.

More information on step-by-step procedures for distributing an MSI file using Active Directory is available at

http://support.microsoft.com/?kbid=321713.

More information on Active Directory and its setup documentation is available at

http://www.microsoft.com/windowsserver2003/technologies/directory/activedirectory/default.mspx

Systems Management Server

SMS is another tool from Microsoft that you can use to distribute the package. SMS has various features for packaging, distributing, and deploying applications in addition to monitoring them.

More information on step-by-step procedures for distributing a package using SMS is available at

http://support.microsoft.com/default.aspx?scid=kb;en-us;842844.

Installation of the Application

After packaging an application and distributing it to the local system, the next step is to install the application. You can deploy or install an application in the following ways:

- **Using Windows Installer.** Windows Installer is part of the Windows operating system and can understand and install MSI files.
- **Using Wise or InstallShield.** Wise and InstallShield provide their own proprietary scripts that can be understood by Windows as application-installable packages.

For more information on deploying an application, refer to the “Development Considerations for Deployment” section in the “Closing the Developing Phase” chapter of the build volumes (Volume 2, Volume 3, and Volume 4).

Creating Operations Procedures

The major role of the operations team is to maintain and monitor the application continuously after deploying the application. This section discusses the objectives, roles, and responsibilities of operations and how the operations team validates the project using various plans created during the Planning Phase.

Team Objectives, Roles, and Responsibilities

The operations team is represented on the project team throughout the project by the Release Management Role. As the Stabilizing Phase nears completion, the teams begin to transfer their activities and responsibilities to a deployment team. The operations team takes the responsibility, assisted by certain members of the development team, to help with deployment and stabilization.

Project Validation

The operations team validates the system's environment, equipment, software, and configurations as well as the following plans:

- **Disaster recovery plan**. During the Planning Phase, the team creates a disaster recovery plan to maintain the migrated applications in case of crisis. During the Deploying Phase, the team reviews the document and validates and updates its contents. This plan should contain the potential risks and resolutions, the persons responsible for each component, and the methods for notifying management of any problems that occur.
- **Business contingency plan**. During the Planning Phase, the team creates a business contingency plan, establishing the course of action to take in case any business activity halts. During the Deploying Phase, the operations team reviews and validates the contents of the business contingency plan.
- **Security plan**. During the Planning Phase, the team creates a security plan. During the Deploying Phase, the operations team upgrades and confirms that all security procedures are in place. The team ensures that all permissible personnel are aware of the security standards and rules to which the project adheres.

Procedural Checklist

The operations team is responsible for maintaining and monitoring the migrated application on a daily basis. The team should create a checklist to avoid any potential issues with the migrated application. The team should consider the following factors when preparing the checklist:

- Dependency on the external tools or libraries
- Service packs, drivers, and new versions of hardware and software
- Data integrity and backups
- Capacity planning

Deploying the Solution

In deploying the solution to the production environment, you should consider the following factors:

- Transition to deployment
- Configuration management
- Readiness to deploy
- Deployment considerations
- Core solution

The following subsections describe the preceding factors in detail. The information provided here will assist you in deploying the migrated application to the production environment.

Transition to Deployment

The project team continuously updates and revises the deployment plan during the Developing and Stabilizing Phases. The Release Manager must review and revise the deployment plan and confirm that the development team has accomplished the following tasks:

- The deployment strategy is reviewed and approved.
- Setup, installation, configuration, test, operation, and support procedures are available, reviewed, and approved.
- Deployment procedures are documented, reviewed, and approved.
- Hardware and software components are available and tested.
- Ownership is transferred to the operations team.

It is vital that the project team continues to monitor the solution after deployment until the transfer of ownership is finalized. This allows the team to establish baseline performance records to track the performance of the application.

Configuration Management

Configuration management serves the same purpose for a deployed solution that revision control serves for source code under development. Operations personnel can use this configuration information to control the deployment environment. When deploying the migrated solution, it is vital for the Release Manager to provide all information required by the operations team for tracking configuration information.

Readiness to Deploy

The team should determine an application's readiness for deployment after reviewing the test results and various checklists prepared during the testing activities in the Stabilizing Phase. The checklists are not intended to be decision-making devices. If the team finds any specific issue not on the checklist, it probably indicates that the application is not ready for deployment.

Deployment Considerations

A potential approach to take when going live is to review each layer of the architecture and run through the critical considerations. The considerations should be relevant to the database tier, the application and business layer, the presentation layer, and the general technical and business issues. The team should prepare a checklist with all the considerations for each of these layers.

Note For more information on deploying an application, refer to the "Development Considerations for Deployment" section in the "Closing the Developing Phase" chapter of the build volumes (Volume 2, Volume 3, and Volume 4).

Core Solution

Most solutions include a number of code components as well as infrastructure components that provide the framework or backbone for the entire solution. These components do not represent the solution from the perspective of a specific set of users or site, but the deployment of the solution to sites or users generally depends on this core set of features or technology.

Interim Milestone: Core Technology Deployed

After the deployment of the core technology, all components that are part of the solution and its dependencies are in place, running properly, and sufficiently stable to move forward with site deployment components.

Interim Milestone: Site Deployments Complete

At the completion of this interim milestone, all targeted users have access to the solution. Each site owner has signed off that his or her site is operating, although there may be some issues. Some sites might have to be revisited based on customer feedback. To reach the Site Deployments Complete interim milestone, the team makes a concentrated effort to finish deployment activities and stabilize and close the project. This interim milestone is applicable only for projects that involve client-side deployments.

Stabilizing the Deployment

This section describes the activities that the project team needs to carry out for stabilizing the deployment before transferring ownership to the operations team. This section describes the steps to be considered for stabilizing the deployment, disengaging from deployment activities, and transferring the deployment to the operations team.

The Release Management lead assigns deployment tasks to the team. The team must review the project status and test results and update the deployment plan periodically. The team must also create task-based procedures that ensure successful deployment of the application.

The information gathered during the previous phases will help the team identify tasks that must be completed before transferring the application to the operations team.

Stabilizing the solution is a joint venture between the project and the operations teams. Stabilizing usually begins in the later part of the Stabilizing Phase and continues throughout the Deploying Phase as each solution component is put in place. It is important to include time in the master schedule for deployment stabilization because it gives both teams an opportunity to fine-tune the solution, resolve any issues that might arise, and monitor the solution to ensure that it continues to work properly after going live. It also allows the project team to assist the operations team after the transfer of ownership.

It is expected that some issues will arise with the various site deployments. These issues should be continuously tracked and resolved. At the Deployment Stabilized interim milestone, the customer and team agree that the entire solution is operating satisfactorily.

Disengagement Considerations

The team can find it difficult to close the project because of the ongoing issues that will surface after deployment. For this reason, the team needs to clearly define a completion milestone and criteria for the deployment instead of attempting to reach a point of absolute finality. Usually, one or two members will remain on the project for a short period of time to follow through on any problems or undocumented instructions. At this point, disengaging from the project includes transferring operations and support functions to permanent staff.

The Quiet Period

The period between the Deployment Stabilized interim milestone and Deployment Complete Milestone is referred to as a quiet period. The purpose of the quiet period is to measure how well the solution is working in typical operation and to establish a baseline for understanding how much maintenance will be required to run the solution. Organizations using Microsoft Operations Framework (MOF) will measure the number of incidents and downtime and collect performance metrics for the solution.

Transferring Ownership to the Operations Team

This section describes the steps for transferring the solution and the respective tasks and responsibilities of the project and operations teams. At this point, the project team must complete its final tasks, gather all produced collateral, and transfer the solution and the technology to the operations team. Relevant documents related to the migrated application must be updated to reflect changes in functionality and operation as a result of the migration. Some of these documents are:

- Deployment diagrams.
- Event-sequence charts from actual observations.
- Test plan including test reports.
- Reliability plan.

- Security plan.
- Backup plan to prevent loss of data.
- Plan for analyzing system performance and solution usage.
- Plan for log handling and other administrative tasks.
- Disaster recovery plan.
- Business contingency plan.
- Training information.

Transferring the Solution

Transferring the solution is the responsibility of both the project and operations teams. The project team will close any open issues and hand off maintenance activities to the operations team. The operations team will perform the specified activities and validate all collateral and equipment received from the project team. The following subsections describe the activities of both the project team and operations team when transferring the solution.

Project Team Tasks

The project team's closing tasks include:

- Verifying that the operations team has the proper resources to maintain and manage the new solution.
- Confirming the final transfer of the knowledge base to the operations team.
- Reviewing operations logbooks and ensuring that the procedures are being properly performed.
- Reviewing configuration management data for the deployed solution to ensure that any changes in deployed components are accurately reflected.

1. Reviewing Help documents for the migrated application.

- Reviewing training materials for the migrated application.
- Confirming that all project sign-offs are correct. After the signatures have been obtained, the transfer of ownership starts, and the operations team is solely responsible for the solution.

Operations Team Tasks

During the transition period, the operations team will work closely with the project team and complete the following activities:

- Activating all reporting systems.
- Validating and publishing all collateral.
- Validating and publishing the knowledge and databases.
- Reviewing training materials provided by the project team.

After these activities are complete, the project team transfers the solution and documentation ownership to the operations team.

Interim Milestone: Deployment Stabilized

At this milestone, the solution is deployed and has reached a quiet period where the operation of the solution is under control, users have started to receive business value from the solution, and the project is moving toward closure.

Closing the Deploying Phase

Closing the Deploying Phase requires completing a milestone approval process. The team needs to document the results of the different tasks it has performed to sign off on completion of the phase.

The deliverables checklist for this phase includes:

- Final versions of all solution documents, code, and test cases.
- Training documentation.
- Service level agreements (SLAs).
- Operations and support information.
- Customer and user satisfaction data.
- Project close-out report.
- Definition of next steps.
- Updated risk management.
- Milestone review report.
- Team member project progress report.
- Team lead project progress report.

Key Milestone: Deployment Complete

Chapter 2: Operations

This chapter deals with the topic of operating your UNIX-to-Microsoft® Windows® migration solution after the migration project has been completed. Much of the relevant guidance can be found in existing documentation on Windows operation. The guidance provided here supplements this documentation and addresses three areas:

- Aspects of the Windows environment that will be modified in some respect as a result of the migration.
- Aspects of migrated solutions that are not covered by existing guidance.
- Interoperation issues.

After the application has been successfully deployed, the operations team is responsible for maintaining the solution on a continuous basis, which allows the team to proactively deal with problems. The team performs all the daily maintenance procedures and provides low-level routine change management support for the solution. The success of ongoing operations depends heavily upon the "people and process" elements as well as the technology.

In this guide, the guidance for completing the technical elements of the migration project is organized according to the Process Model phases outlined in Microsoft Solutions Framework (MSF). But the operations guidance (the “people and process” elements) offered in this chapter is organized according to Microsoft Operations Framework (MOF), which is a complementary framework to MSF.

The development and deployment of an IT solution typically involves two teams: the project team and the operations team. The project team is mainly responsible for the MSF Planning, Developing, and Deploying Phases of the migration. MSF provides a flexible and scalable way to plan, design, develop, and deploy successful IT solutions. In contrast, the operations team is permanent and is responsible for the solution’s daily operations and management. MOF is designed to guide the operations teams to achieve mission-critical system reliability, availability, supportability, and manageability of IT solutions.

The two frameworks are complementary, minimizing the time to value—that is, the time between recognition of the need and delivery of the service. Consistency of terminology and concepts between the two frameworks also supports the delivery of a high-quality service. The two frameworks are also well integrated. MSF and MOF both include guidance for team roles and processes to ensure a successful deployment into the production environment.

A brief overview of MOF is provided in the next section of this chapter. More information on its basic models and key concepts is provided in the appendix of the *UNIX Migration Project Guide*.

Note More detailed information on MOF is available at http://www.microsoft.com/mof.

Microsoft Operations Framework

Microsoft Operations Framework (MOF) is a collection of best practices, principles, and models. MOF offers a suite of operational guidance in the form of white papers, operations guides, assessment tools, operations kits, case studies, templates, support tools, and services. MOF principles and guidance are organized around three core models: the MOF Process Model, MOF Team Model, and MOF Risk Management Discipline These models are discussed in the following sections. You can use this information to understand the best practices of MOF and apply them to your IT environment to effectively manage and operate the migrated applications.

MOF Process Model

This section provides an overview of the MOF Process Model and its approach to IT operations. The MOF Process Model provides guidelines on how to plan, deploy, and maintain IT operational processes in support of mission-critical service solutions. The MOF Process Model organizes the service management life cycle into four quadrants, with each quadrant having a specific focus and set of tasks that are carried out through its corresponding set of service management functions (SMFs). (MOF defines more than 20 SMFs.) The key components of the MOF Process Model are:

- Quadrants
- Operations management reviews
- Service management functions

Figure 2.1 depicts the MOF Process Model.

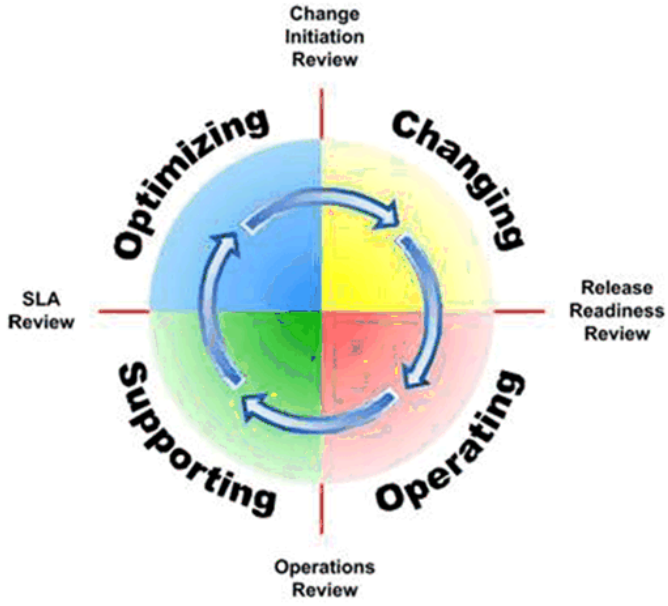

Figure 2.1. MOF Process Model and its quadrants

The following sections explain these components in detail.

Quadrants

Typical operational activities are defined and grouped into quadrants, each of which is focused on a particular set of processes and tasks. The entire MOF Process Model can be divided into four quadrants, as described as follows.

Changing Quadrant

The Changing Quadrant describes processes, responsibilities, reviews, and best practices that help organizations manage changes to their IT infrastructure. MOF identifies three key sets of activities, or SMFs, within this quadrant:

- Change Management
- Configuration Management
- Release Management

In addition to these SMFs, MOF recommends two specific milestone reviews, the Change Initiation Review and the Release Readiness Review, as part of the change management process. These reviews ensure that the new solutions or other significant changes align with the business and the operational requirements for operability.

Note More information on each of these components of the Changing Quadrant is available at http://www.microsoft.com/technet/itsolutions/cits/mo/mof/mofchange.mspx.

Operating Quadrant

The Operating Quadrant is the collection of processes and IT functions dedicated to the ongoing maintenance, monitoring, control, and protection of IT infrastructure or applications. MOF identifies seven key sets of activities, or SMFs, within this quadrant:

- Directory Services Administration
- Job Scheduling
- Network Administration
- Security Administration
- Service Monitoring and Control
- Storage Management
- System Administration

In addition to these SMFs, MOF recommends the periodic completion of an Operations Review in order to permit IT service managers to analyze their performance and processes.

Note More information on each of these components of the Operating Quadrant is available at http://www.microsoft.com/technet/itsolutions/cits/mo/mof/mofopman.mspx.

Supporting Quadrant

Activities and processes that are performed to resolve user- and system-generated queries, issues, or problems are in the domain of the Supporting Quadrant. MOF identifies three key sets of activities, or SMFs, within this quadrant:

- Incident Management
- Problem Management
- Service Desk

In addition to these SMFs, MOF recommends the periodic completion of a Service Level Agreement (SLA) Review and an analysis of past performance and future commitments.

Note More information on each of these components of the Supporting Quadrant is available at http://www.microsoft.com/technet/itsolutions/cits/mo/mof/mofsupport.mspx.

Optimizing Quadrant

The Optimizing Quadrant encompasses processes and IT functions dedicated to planning and implementing enhancements to the IT environment through a continuous cycle of process improvement. Recommendations originating in the Optimizing Quadrant generally are reflected as changes in the IT infrastructure and are instituted through the change management process, which is a part of the Changing Quadrant.

MOF identifies eight key sets of activities, or SMFs, within this quadrant:

- Availability Management
- Capacity Management
- Financial Management
- Infrastructure Engineering
- IT Service Continuity Management
- Security Management
- Service Level Management
- Workforce Management

In addition to these SMFs, recommended changes will be implemented through change management, initiated by a Change Initiation Review.

Note More information on these components of the Optimizing Quadrant is available at http://www.microsoft.com/technet/itsolutions/cits/mo/mof/mofoptimize.mspx.

Operations Management Reviews

The operations management reviews (OMRs) are a unique feature of MOF; they contain reviews of operational activity for efficiency and effectiveness. MOF makes these reviews an explicit part of the Process Model and provides detailed guidance on how to conduct them. These OMRs are specifically labeled on the Process Model diagram because they warrant senior management attention and can be used as a regularly reported "health check" on the state of the operations activity. The operations management reviews are:

- Release Readiness Review
- Operations Review
- SLA Review
- Change Initiation Review

The Process Model incorporates two types of management reviews: release-based and time-based. Two of the four OMRs—Release Readiness Review and Change Initiation Review—are release-based and occur at the initiation and final installation of a release into the target environment, respectively. The remaining two OMRs—Operations Review and SLA Review—occur at regular intervals to assess the internal operations as well as performance against customer service levels.

Service Management Functions

Service management functions (SMFs) are the underlying processes and activities within each MOF quadrant that support the mission of service for that quadrant. These SMFs are at the core of the MOF Process Model. Although all of the SMFs are cross-functional (and cross-quadrant) in nature, each SMF is assigned a "home" or "primary" quadrant that aligns the functions performed with the mission of service for that quadrant. (The specific SMFs applicable to each quadrant of MOF are listed in the preceding sections.) This natural alignment between each quadrant and its corresponding SMFs allows the IT manager to see all the key SMFs that are required to effectively run the operations environment.

MOF Team Model

The MOF Team Model organizes the activities of IT operations into seven distinct role clusters that represent areas, or functional roles, within IT operations.

The seven role clusters of the Team Model define the general categories of activities and processes, common ways to identify roles and responsibilities, and common goals for each specialist function team required for a successful operations management organization.

Table 2.1 describes the Team Model quality goals and lists the key functional roles responsible for achieving them.

Table 2.1. MOF Team Quality Goals and Team Roles

Quality Goal	Team Role
Well-developed release and change management processes and accurate inventory tracking of all IT services and systems.	Release
Portfolio of business-aligned IT services.	Service
Efficient management of physical environments and infrastructure tools.	Infrastructure
High-quality, cost-effective customer support and a service culture.	Support
Predictable, repeatable, and automated day-to-day system management.	Operations
Protected corporate assets, controlled access to systems and information, and proactive planning for emergency response.	Security
Efficient and cost-effective, mutually beneficial relationships with service and supply partners.	Partner

MOF Risk Management Discipline

The MOF Risk Management Discipline applies proven risk-management techniques to the challenges that operations staff members face every day. The MOF Risk Management Discipline offers greater value than many other risk management models through its key principles, consistent terminology, and structured and repeatable six-step process. Following is a brief introduction to the six steps of the risk management process.

- **Identify.** Risk identification allows you to identify risks so that the operations staff becomes aware of potential problems.
- **Analyze and prioritize.** Risk analysis and prioritization enables operations to analyze and prioritize the possible risks and commit resources to manage them.
- **Plan and schedule.** Risk planning takes the information obtained from risk analysis and uses it to formulate strategies, plans, change requests, and actions. Risk scheduling ensures that these plans are approved and then incorporated.
- **Track and report.** Risk tracking monitors the status of specific risks and the progress in their respective action plans. Risk reporting ensures that the operations staff, service manager, and other stakeholders are aware of the status of top risks and the plans to manage them.
- **Control.** Risk control is the process of executing risk action plans and their associated status reporting. Risk control also includes initiating change control requests when changes in risk status or risk plans could affect the availability of the service or SLA.
- **Learn.** Risk learning formalizes the lessons learned and uses tools to capture, categorize, and index that knowledge in a reusable form that can be shared with others.

Operating a Mixed Environment

This section explains different operations mechanisms that you can use when managing a migrated application post-deployment. The section also provides information about tools and software available for the preceding purpose and how they can be used for managing application environments.

For many organizations, a migrated application functions in an environment containing a mixture of UNIX and Windows operating systems. Such organizations need some interoperation between the two operating systems. To some extent, interoperation is also required where Interix and Windows systems coexist.

The following are the main topics that you will need to address in mixed environments:

- Windows-to-UNIX connectivity
- User authentication and authorization
- Resource and data sharing
- Tools and scripts

Windows- to-UNIX Connectivity

In a mixed live environment in which users and support staff access applications running on Windows-based and UNIX-based computers, you will require tools to connect between the operating systems. The client software usually available on Windows desktops provides character or graphical (X Windows) access to UNIX applications. Modern terminals access the UNIX systems either from the command line or from a graphical user interface (GUI).

Command-line access tools include telnet, Berkeley **r** access commands (**rsh**, **rexec**, and **rcp**), and secure shell (**ssh**), which have the following features:

- The telnet client allows the user to operate in the multiuser environment provided by the host, with access provided by the telnet server.
- The **rsh** command allows the user to run commands remotely from a local environment.
- The **ssh** command provides extra authentication and encryption features. Its functionality is similar to the **rsh** command, except that **ssh** can be made very secure.

The GUI access tool is X Windows, which has the following functionality:

- An X Windows server provides a graphical interface to local or remote applications because it uses a TCP/IP connection to the UNIX-based computer.

Developers and users of UNIX applications typically use one or more of these access methods. The access method that is used affects both the migration strategy and the migration process itself.

Users require UNIX connectivity when the migrated application continues to use UNIX-style interfaces, such as in some migrations to the Interix environment.

The Windows operating system includes a basic telnet client and supports the **rsh**, **rexec**, and **rcp** client commands.

With an X Windows server on the desktop, developers can also use a GUI to connect to UNIX applications. Windows operating systems do not provide X Windows. For X Windows connectivity, developers need a third-party X Windows server.

The following sections describe the main connectivity solutions available.

Terminal Emulation and Command-Line Connectivity

UNIX developers use the tools described in this section to access UNIX development servers from the command line. If the UNIX application has a command-line interface, these tools are also helpful to users.

Windows Telnet Client

The Windows telnet client provides basic command-line access to UNIX systems. Its command syntax is identical to that of the telnet clients found on UNIX operating systems. The Windows telnet client runs in a window. Users can copy text between the UNIX command line and Windows-based applications. Users run the client from the command prompt or use the **Run** option on the **Start** menu.

The Microsoft HyperTerminal application also includes a telnet client. HyperTerminal offers more configuration options than the command-line telnet client. Terminal types for the client range from ANSI to VT52 to VT100. It is simple to copy text between HyperTerminal and Windows-based applications.

A third-party telnet client may be required for command-line tools that use terminal characteristics beyond those provided by HyperTerminal, such as function keys or full-screen operation.

Windows Telnet Server

Developers who need to connect from UNIX-based to Windows-based servers can install the telnet server included in the Microsoft Windows Server™ 2003 operating system.

Remote Shell (rsh), Remote Copy (rcp), and Remote Execute (rexec)

Using the Berkeley **r** commands for remote access to UNIX servers simplifies the process. Authentication occurs only once on the logon to the UNIX system, after which a user does not need to log on again to connect to other systems. Because of this, it is easy to run remote shells (**rsh**) and remote commands (**rexec**) and to copy files between remote computers (**rcp**).

The remote clients are included in Windows. The syntax for the Windows commands is:

```
rsh [Host] [-l UserName] [-n] [Command]
rcp [{-a | -b}] [-h] [-r] [Host][.User:] [Source] [Host][.User:]
[Path\Destination]
rexec [Host] [-l UserName] [-n] [Command]
```

On Windows-based clients, the *UserName* parameter is required if the names on UNIX and Windows are different.

The *Windows 2000 Resource Kit* includes an **rsh** tool, **Rshsvc.exe**.

Secure Shell (ssh)

The secure shell provides extra authentication and encryption features. It is often used to replace telnet and the Berkeley **r** access commands where security is an issue. The **ssh** command functionality is similar to the **rsh** command except that **ssh** can be made very secure. (Client and user information is not revealed to third parties.)

Windows operating systems do not include a secure shell client or server. However, third-party products do provide the **ssh** functionality.

GUI Access Tools

Users who require GUI access to a UNIX-based server from a Windows-based desktop usually use an X Windows server.

X Windows

Windows operating systems do not include an X Windows server. The Interix environment includes X Windows clients, which can be used to develop applications that use an X Windows user interface. However, Interix does not include X server.

Third-party products that provide Windows-based X Windows servers are well-integrated with the Windows desktop. Users of third-party products can copy text and graphics between X Windows and Windows. Third-party Windows-based X Windows servers include:

- **Hummingbird Exceed.** More information on Hummingbird Exceed is available at http://www.hummingbird.com.
- **MKS Toolkit**. More information on MKS Toolkit is available at http://www.mks.com.
- **WRQ Reflection X**. More information on WRQ Reflection is available at http://www.wrq.com.

These products provide support for most X Windows standards. Each product also includes extra information that will be useful. For example, MKS Toolkit also includes more than 300 UNIX commands that can be used within Windows.

Remote Desktops

Windows operating systems handle multiple users in a different way than UNIX does. A single user at a time logs on to a Windows-based computer. However, using Windows 2000 Terminal Services is analogous to using X Windows on UNIX. By using Terminal Services, a Windows-based server can provide a desktop to remote users on clients spread over a network. This is often referred to as a "thin client" because the client only has to handle input and output while the application runs on the server.

For UNIX-to-Windows connectivity, UNIX and Linux implement the RDP protocol used by Terminal Services.

To connect to a Windows-based server from a UNIX client using a remote desktop, implement the Citrix MetaFrame. The Citrix MetaFrame has clients for a wide range of operating systems, including UNIX and Linux. Also, an X Windows interface to UNIX is required.

Note More information on Citrix MetaFrame is available at http://www.citrix.com/products/.

User Authentication and Authorization

In any heterogeneous network, such as a network being migrated to Windows, users need to work across systems, and the systems need to interoperate. Your migration project needs to take into account the differences between the UNIX and Windows security models. Security-related questions that were addressed as part of planning need to be implemented here. For example:

- Number of user databases that administrators need to manage.
- Mode of managing user security on shared resources, such as network file system (NFS).

Resource and Data Sharing

Migration from UNIX to Windows requires the two environments to share data and resources. In the simplest case, UNIX-based servers migrate files to Windows-based servers. On the other hand, UNIX-based servers and Windows-based servers share data and resources, such as printers, during and after migration.

There are some less-integrated methods for transporting files between systems, such as file transfer protocol (FTP) and removable media (for example, tape). These methods are not considered in this guide because they do not provide interoperability. However, removable media and FTP can be very useful during the migration. Tape is particularly effective for transferring volumes of data that are too large to feasibly transfer across a network.

Developers or application users must be able to use the files that they need in their target environment (Windows or Interix) transparently. They do not need to know whether the files are on a UNIX-based server or on a Windows-based server. However, UNIX-based systems and Windows-based systems use different network protocols, functionality, and naming conventions for file sharing.

Tools and Scripts

Your system and application administrators will usually want to use tools and scripts to manage your migrated application. With migrations to Microsoft Interix and Microsoft Win32®, you have the option to migrate any management tools that were used previously in your UNIX environment.

In a Win32 migration, you can also use some of the Windows-based management tools, such as Microsoft Systems Management Server (SMS).

It is likely that you will consider porting your UNIX management tools to Interix because it gives a UNIX-like environment. Consider porting the tools if your application has been migrated to Win32 and continues to use the same configuration files and scripting tools as it did on UNIX. In this case, Windows Services for UNIX 3.5 provides the tools necessary to port Perl and UNIX shell scripts to Win32, including those employing standard UNIX tools such as **awk**, **sed**, and **grep**. These topics are covered in detail in Chapter 2, "Developing Phase: Process Milestones and Technology Considerations" of the respective build volumes (Volume 2, Volume 3, and Volume 4) of this guide.

Operating in the Windows Environment

After successfully deploying the migrated applications in the Windows environment, you should follow certain key activities to operate and maintain the deployed applications. These key activities are:

- Service monitoring and control.
- Exception monitoring.
- Notification.
- Event monitoring.

The following subsections describe these activities in detail.

Note More information on operating applications in a Windows environment is available at http://www.microsoft.com/technet/prodtechnol/windowsserver2003/operations/default.mspx and http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/default.mspx.

Service Monitoring and Control

This section provides an overview of different monitoring mechanisms that can be adopted to monitor the migrated applications in the operating stage.

The monitoring process for an application involves:

- Monitoring the health and status of enterprise systems.
- Alerting administrators to problems and consolidating the monitoring data in a single place for ease of administration.

The monitoring tools can monitor individual servers or network components, or they may focus on application services like e-mail, transaction processing, or Web services.

Predictive event management is an essential goal for Windows. Instead of reacting to failures, these services report potential errors and failures before they happen so that corrective action may be taken. For example, a predictive exception monitoring tool might notice the increasing frequency of disk write errors on a particular disk and may notify the system manager that the disk is likely to fail in the near future. The specifics of prediction will necessarily vary from site to site.

The Web-Based Enterprise Management (WBEM) initiative is an effort to provide a unifying mechanism for describing and sharing management information. WBEM provides a general mechanism that allows any application or service to supply the raw data needed for prediction.

Note More information on WBEM is available at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwmi/html/wmioverview.asp.

A recommended tool for service monitoring and control for Windows is Microsoft Operations Manager (MOM) 2005. It provides:

- Comprehensive event management.
- Proactive monitoring and alerting.
- Reporting and trend analysis.

Note More information on MOM is available at http://www.microsoft.com/mom/techinfo/productdoc/default.mspx.

Exception Monitoring

Exception monitoring tools monitor conditions that fall outside a predefined range of conditions. An administrator would want to know when something unusual happens. Exceptions are usually generated when the amount of a system resource like free disk space falls below a preset threshold or when a monitored device reports a failure or anomaly. Exceptions may be advisory (for example, "This disk is down to 25 percent free space") or they may be critical (for example, "The drive at SCSI ID 3 just went offline"). The response to any exception depends on its severity.

There are several potential sources of exception data. The Microsoft Windows Server 2003 event log records notifications from applications, drivers, and system services, and the log can be used as the basis for generating exceptions.

Note More information on how to diagnose system problems with Event Viewer in Windows Server 2003 is available at http://support.microsoft.com/default.aspx?scid=kb;en-us;302542&sd=tech.

Simple Network Management Protocol (SNMP) and performance monitor tools are also available for the same. SNMP is a popular protocol for network management. It is used to collect information from, and to configure, network devices, such as servers, printers, hubs, switches, and routers on an Internet Protocol (IP) network. Windows Server 2003 provides SNMP agent software that works with third-party SNMP management software to monitor the status of managed devices and applications. SNMP managers can generate exceptions when they receive traps from their managed devices, and BackOffice components like Microsoft Exchange and Microsoft SQL Server™ can trigger exceptions when service-specific events occur, for example, when a remote mail server does not respond to messages in a predefined interval.

System Monitor is a tool used to track a range of processes and give a real-time graphical display of the results on a Windows Server 2003 system. This tool can assist you with the planning of upgrades, tracking of processes that must be optimized, monitoring results of tuning and configuration scenarios, and the understanding of a workload and its effect on resource usage to identify bottlenecks. Navigate to the performance icon in the administrative tools folder in Control Panel to open the monitor. Alternatively, you can open it from the **Start** menu or by typing **perfmon.msc** in the **Run** dialog box.

System Monitor in Windows Server 2003 supports an **Alert** view, in which, the designated parameters are continually monitored and exception messages are generated when any parameter goes out of limits. Administrators can also use WSH or any other supported scripting mechanism to write flexible scripts that monitor the system and send exception messages when needed.

Exception monitoring in UNIX is usually done by administrators. They can write scripts to monitor their application. Scripts are often not portable among variants of UNIX, and third-party applications usually require the purchase of their own proprietary monitoring components.

Notification

Notification is the process by which administrators or automated systems are notified that some condition has gone out of bounds. These messages can travel on a variety of paths, the most common of which are:

- Alphanumeric pagers.
- E-mail.
- On-screen alerts.
- Adding entry to event log.

The *Windows Server 2003 Resource Kit* includes tools for sending exception alerts through e-mail from a command line or script. These tools can be used with the built-in exception capacity

of the event log and System Monitor. Alphanumeric paging messages can be sent using freeware or commercial third-party tools that dial into a paging service and upload the message.

In addition, BackOffice components and many third-party applications include their own notification mechanisms. For example, Microsoft Exchange Server can be used to send e-mail, display on-screen alerts, or route exceptions to an external application. These capabilities are in addition to, and not a replacement for, the standard system notification paths.

Both UNIX and Windows support a variety of third-party notification products. These products usually combine exception monitoring with flexible rule sets that determine who gets which notification messages.

Event Monitoring

A network services and consulting firm provides dial-up Internet access for government customers who, for security reasons, cannot have dedicated local access. The organization depends on its servers running Windows NT® and UNIX for its own internal network, including file and print services, the Web page of the organization, and its sales database.

The network administrator and data center managers each carry a two-way pager. The UNIX systems may run custom-written shell scripts that monitor various conditions and send an e-mail to the address of the pager when a significant event occurs. For example, the daily backup script will send a success message to the regular mail accounts of the administrators, but will send failure messages directly to the pager of the on-call administrator.

The Windows NT-based systems use pager notification through System Monitor and Exchange Administrator. Both of these programs allow an external program to be launched when a monitored parameter goes out of limits. Exception messages are routed to the pagers by a small command-line mail-sending tool. Windows NT-based servers also use **ksh** shell scripts (using the Interix UNIX-compatibility toolkit) to monitor some of the same conditions as the UNIX servers. For example, the Windows NT® backup script is largely identical to the UNIX script.

The notification system allows administrators to receive continual updates of ongoing processes like backups, as well as instant notification of problems with the network or servers. In many cases, problems can be resolved before the customer encounters them.

Advantages of a Migrated Application to .NET

This section describes the various features available with .NET Framework that help in maintaining or managing the application after it is migrated from UNIX to Windows. This section acquaints you with .NET features and their usage in the operations activities of the migrated application. After migrating your application to Windows Services for UNIX 3.5 or Win32, you can take advantage of the new functionality exposed by the common language runtime (CLR) and reuse existing components from the managed code that you develop.

The interoperability features of .NET enable you to work with existing unmanaged code (that is, code running outside the CLR) in Component Object Model (COM) components as well as Microsoft Win32 DLLs. It also allows the use of managed components from the unmanaged, COM-based code.

The .NET Framework provides the following three forms of interoperability:

- **Platform Invoke (P/Invoke)**. This feature allows the use of managed languages, such as C# and Microsoft Visual Basic® .NET, to call libraries within standard Microsoft Windows DLLs (such as the Win32 API) or custom DLLs.
- **It Just Works (IJW)**. This feature allows the use of Managed Extensions for C++ to directly call libraries within standard Windows DLLs. IJW provides C++ programmers with a more straightforward way to call functions in standard Windows DLLs. The appropriate header file should be included for the unmanaged APIs and then linked to the associated import library.

- **COM Interop**. This feature allows the use of managed languages to activate and interact with COM objects through COM interfaces. COM interop simplifies creating and instantiating COM components. You can generate (or acquire) an *interop assembly* to use COM interop and to make early-bound calls to COM components. An interop assembly is an assembly that contains managed types that provide the capability to program indirectly against unmanaged COM types.

Note More information on improving interop performance is available at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag/html/scalenetchapt07.asp.

Advantages of a Migrated Windows Services for UNIX-based Application to .NET

UNIX and Win32 calls cannot be mixed in a single process when building applications using the Interix environment in Windows Services for UNIX 3.5. However, the following alternate ways can be used by which Windows Services for UNIX 3.5 applications can make use of the .NET environment:

- **Encapsulate the Windows Services for UNIX 3.5 application from a Win32 COM object**. This is described is Chapter 7, "Development Considerations for Deployment" of Volume 2: *Migrate Using Windows Services for UNIX 3.5* of this guide. .NET applications can access the resultant COM object through interoperability mechanisms (PInvoke and others), discussed in Chapter 3, ".NET Interoperability" of Volume 4: *Migrate Using .NET* of this guide.
- **Use Web services on Windows Services for UNIX 3.5**. XML Web services provide a clean way to exchange data between Windows Services for UNIX 3.5 and .NET applications without dependencies on the architectural differences between the two platforms. The steps are as follows:
  - Install a Web server on Windows Services for UNIX 3.5. Apache is an open-source Web server that is available for Windows Services for UNIX 3.5. Use third-party components to create a Web service wrapper around your Windows Services for UNIX 3.5 code. Many of these third-party components are available as open source components.
  - .NET applications can now "consume" the Web service. The functionality provided by the Windows Services for UNIX 3.5 Web service can be integrated into any .NET or Win32 application.
- **Use Web services on .NET**. Write a C# code to run the Windows Services for the UNIX 3.5 application and capture its output. Translate this output into XML using the .NET Framework classes. Wrap all of this within a .NET Web service. Other .NET applications can now make use of the functionality of the Windows Services for UNIX 3.5 application through the Web service. The benefit of this approach is that it takes the advantages of the Web services infrastructure into the .NET Framework.

Some of the ways in which C# code can access Windows Services for UNIX 3.5 application are as follows:

- Use command-line invocation through **posix.exe**, which is easy but not very efficient.
- If the UNIX application exposes an RPC server (for example, using ONC RPC), the C# code could make an RPC call to the UNIX-based server. You could build some unsafe C# code that maps a file into memory, and build the UNIX application to map the same file. Shared memory is particularly useful when you need to quickly move large amounts of data between Windows Services for UNIX 3.5 and .NET.

Advantages of a Migrated Win32/Win64-based Application to .NET

An application that has been migrated to Win32/Win64 using the Windows API (application programming interface) can be migrated to .NET in the following ways:

- **Interop Services**. .NET can call Windows API using platform Interop Services, which resides at the **System.Runtime.InteropServices** namespace. The Windows API resides in the

User32.dll, GDI32.dll, and Shell32.dll in the Windows system directory. The Interop Services of .NET work with external DLLs. Therefore, by using the **System.Runtime.InteropServices** namespace in the application, you can make calls to external applications.

Note More information on this is available at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/csref/html/vcwlkPlatformInvokeTutorial.asp.

- **COM objects**. .NET also provides interoperability with COM applications. A COM object can be called from .NET. When the COM object is called from .NET, the runtime generates a runtime callable wrapper (RCW). The RCW acts as a proxy for the unmanaged object and is responsible for handling all interactions between the .NET client code and the COM component.

  Note More information on COM and .NET interoperability is available at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/cominterop.asp and http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconExposingCOMComponentsToNETFramework.asp.

Conclusion

This marks the conclusion of the five volumes of *UNIX Custom Application Migration Guide*. You should now possess sufficient knowledge to migrate UNIX applications to Windows as well as know about all of the related processes that require special attention in order to obtain the best possible solution. The guide has aligned the various activities in the UNIX-to-Windows migration process with the MSF/MOF Process Model in the following volumes:

- **Volume 1: Plan.** This volume described the Windows and UNIX architectures; development of the solution architecture, project plan, and infrastructure; and preparation of the development and testing environments. This volume is aligned with the Envisioning and Planning Phases of the MSF Process Model.
- **Volume 2: Migrate Using Windows Services for UNIX 3.5.** This volume described Windows Services for Unix 3.5, the Interix environment, and architectural differences between Interix and the UNIX environments. It also described the migration activities involved in migrating the UNIX applications to the Interix environment. This volume aligned with the Developing Phase of the MSF Process Model. It provided instructions to developers for the migrating UNIX applications to Windows using Windows Services for UNIX 3.5.
- **Volume 3: Migrate Using Win32/Win64.** This volume provided an overview of the Microsoft Win32 and Win64 architectures and described the steps involved in migrating UNIX applications to the Win32/Win64 environments using the Windows Platform Software Development Kit (SDK). This volume aligned with the Developing Phase of the MSF Process Model.
- **Volume 4: Migrate Using .NET.** This volume described the .NET architecture, configuration of the .NET environment, and the steps involved in migrating UNIX applications to the .NET environment using Microsoft Visual Studio® .NET 2003 and .NET Framework. This volume aligned with the Developing Phase of the MSF Process Model.
- **Volume 5: Deploy and Operate.** This volume described the steps involved in deploying the migration and post-deployment support activities. This volume aligned with the Deploying Phase of the MSF Process Model and operations activities of the MOF Process Model.

This guide has explained the various possible alternatives of migrating UNIX applications to the Windows environment. You can choose the best possible approach based on your business requirements. The guide also provides an assessment tool that can be used for determining the best migration method based on answers to the questionnaire.

Roadmap for Future Migrations

As technology advances, application migrations from UNIX to the Microsoft Windows operating system need to address how to integrate and interoperate with new features and rising

technologies on the Windows platform. This section discusses what to expect for current and future migrations with regard to the following technologies:

- Windows Server System
- Visual Studio .NET 2005
- Microsoft SQL Server™ 2005
- 64-bit solutions
- Microsoft patterns and practices
- High-performance and distributed computing

Windows Server System

The Microsoft Windows Server System provides solutions to a wide-range of application business information management issues, including database management, application deployment and configuration control, business-to-business data interchange, messaging, and workflow management. Following are the some of the products that are part of the Windows Server System:

- **Application Center 2000.** Microsoft Application Center 2000 is a deployment and management tool for high-availability Web applications built on the Microsoft Windows 2000 operating system. It allows deploying, monitoring, and maintaining complex distributed applications consisting of Web sites, Web services, and COM components.
- **SQL Server 2000.** Microsoft SQL Server 2000 is a relational database engine that can store, retrieve, analyze, and manage large amounts of data. SQL Server meets the availability, reliability, and scalability requirements of demanding e-commerce data storage requirements.
- **BizTalk Server 2000.** Microsoft BizTalk® Server 2000 is a suite of tools and services to help you quickly build and deploy data interchange between internal and external Internet applications using W3C-standard Extensible Markup Language (XML) messages.
- **Exchange 2000 Server.** Microsoft Exchange 2000 Server is a messaging infrastructure and data store designed to meet the real-time messaging and collaboration needs of both small and large organizations. Exchange Server 2000 features include instant messaging, real-time conferencing, collaboration, document-centric workflow, calendaring, and contact management. The Exchange data store provides a central repository for e-mail, documents, Web content, and applications.
- **Commerce Server 2000.** Microsoft Commerce Server 2000 is a customizable e-commerce application for creating an online business. Commerce Server 2000 includes many ready-to-use functions, including catalog display, product and customer service management, and order processing.
- **Content Management Server.** Microsoft Content Management Server is a content management system for building, deploying, and maintaining dynamic Web site content for either internal organizational documents or a commercial Web site.
- **Windows 2000 Advanced Server.** Microsoft Windows 2000 Advanced Server is the server operating system for e-commerce and internal business applications.
- **Windows 2000 Datacenter Server.** Microsoft Windows 2000 Datacenter Server is the server operating system for running mission-critical applications and high-volume, real-time transactions.
- **Host Integration Server 2000.** Microsoft Host Integration Server 2000 connects distributed .NET Platform applications to mainframe applications, processes, and data stores. With Host Integration Server, you can access relational data on SQL Server and DB2, and flat-file data on mainframes, AS/400, UNIX, Windows 2000, and Microsoft Windows NT® Server systems.
- **SharePoint Portal Server:** Microsoft SharePoint® Portal Server is a suite of tools and services that can help you create and deploy a central point for publishing business information. It provides document storage and management, content searches, and team collaboration features that are fully integrated with Microsoft Office 2000 applications like Word, Excel®, and PowerPoint®.

- **Systems Management Server.** Microsoft Systems Management Server (SMS) is a suite of tools for handling operating system installation, providing asset management, automating software and hardware inventory, monitoring server health, tracing network topology, and doing remote troubleshooting.

Note More information on Windows Server System is available at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vsent7/html/vxconnetenterpriseservers.asp.

Visual Studio .NET 2005

Visual Studio .NET 2005 is a complete set of development tools for building ASP .NET Web applications, XML Web services, desktop applications, and mobile applications. Visual Basic, Visual C++, Visual C#, and Visual J# all use the same integrated development environment (IDE), which allows them to share tools and facilitates in the creation of mixed-language solutions. It has enhanced features as compared to Visual Studio .NET 2003 in Office application and Web application development. Visual Studio 2005 Team System is a productive, integrated, and extensible software development life-cycle tools platform that helps software teams by improving communication and collaboration throughout the software development process.

Note More information on about the new features of Visual Studio .NET 2005 is available at http://msdn2.microsoft.com/library/88fx1xy0.aspx.

SQL Server 2005

SQL Server™ 2005 is a comprehensive database platform providing enterprise-class data management with integrated business intelligence (BI) tools. The SQL Server 2005 database engine provides more secure, reliable storage for both relational and structured data, enabling you to build and manage highly available, highly performing data applications. The close integration of SQL Server 2005 with Microsoft Visual Studio, the Microsoft Office System, and a suite of new development tools, including the Business Intelligence Development Studio, helps a developer or database administrator reduce development and administrative time.

The features of SQL Server 2005 include:

- **.NET Framework Integration.** With the release of Microsoft SQL Server 2005, database programmers can now take full advantage of the Microsoft .NET Framework class library and modern programming languages to implement functionality within the server. Using common language runtime (CLR) integration, you can code your stored procedures, functions, and triggers in the .NET Framework language of your choice. Microsoft Visual Basic .NET and the C# programming language both offer object-oriented constructs, structured exception handling, arrays, namespaces, and classes.
- **Transact-SQL and Managed Code.** You can write stored procedures, triggers, and user-defined functions using T-SQL or a programming language that is compatible with .NET Framework, such as Visual Basic .NET or C#.
- **Web Services.** In SQL Server 2005, you can develop XML Web services in the database tier, making SQL Server an HTTP listener. This provides a new type of data access capability for applications that are centralized around Web services.
- **ADO.NET.** The next version of ADO.NET supports query change notifications to multiple active result sets (MARS). ADO.NET evolves dataset access and manipulation to achieve greater scalability and flexibility.
- **SQL Management Objects.** The SQL Management Objects (SMO) model is the management object model for SQL Server 2005. SMO represents significant design and architectural improvements for the SQL Server management object model. SMO is the primary tool for developing database management applications using .NET Framework.
- **XML Technologies.** XML has become a common format for storing and transferring data and is a popular choice for marked-up, structured, or semi-structured information. Microsoft SQL Server 2000 supports the use of XML through Microsoft SQLXML, which allows you to convert relational data to an XML format and store XML data in relational tables.

- **New Application Framework.** SQL Server 2005 introduces a new SQL Server application framework, Service Broker. Service Broker is a distributed application framework that provides reliable asynchronous messaging at the database to database level.

Note More information about the new features of SQL Server 2005 for Visual Studio .NET 2005 is available at http://microsoft.com/sql/ .

64-Bit .NET Framework

The 64-bit version of.NET Framework 2.0 enables the .NET Framework platform, tools, and applications to run on 64-bit workstations and servers, which provides increased performance and scalability by addressing more memory (16 terabyte versus 4 GB), processing more data (64 versus 32 bits) per clock cycle, and performing faster numeric calculations. The 64-bit version of the .NET Framework 2.0 allows you to take full advantage of the underlying 64-bit hardware platform without having to deal with differences between the 32-bit and 64-bit hardware platforms. In most cases, applications developed using the 32-bit .NET Framework can be ported to the 64-bit version of.NET Framework and be executed as 64-bit native applications without any source code modifications except the floating point code and the native code. The 64-bit version of .NET Framework 2.0 runs on Windows Server 2003 SP1 64-bit or Windows XP 64-bit, both of which support the two mainstream 64-bit processor architectures.

Note More information on the 64-bit version of .NET Framework is available at http://msdn.microsoft.com/netframework/programming/64bit/.

Microsoft Patterns and Practices

Patterns and practices are recommendations by Microsoft for architects, software developers, and IT professionals responsible for delivering and managing enterprise systems on the Microsoft platform. Patterns and practices are available for both IT infrastructure and software development topics. Patterns and practices are based on real-world experiences that go far beyond white papers to help enterprise IT professionals and developers quickly deliver sound solutions. This technical guidance is reviewed and approved by Microsoft engineering teams, consultants, Product Support Services, and by partners and customers. Organizations around the world have used patterns and practices to:

- Reduce project cost.
- Exploit engineering efforts by Microsoft to save time and money on projects.
- Follow Microsoft recommendations to lower project risks and achieve predictable outcomes.
- Increase confidence in solutions.
- Build solutions on proven recommendations from Microsoft for total confidence and predictable results.
- Deliver strategic IT advantage.
- Gain practical advice for solving business and IT problems today, while preparing companies to take full advantage of future Microsoft technologies.

There are three types of patterns and practices:

- **Guides.** Guides consist of written guidance to obtain a detailed understanding of technical problem domains and engineering practices. Microsoft patterns and practices guides cover topics such as patterns, application architecture, integration, performance, and security.
- **Reference Implementations.** Reference implementations are executable sample applications that demonstrate patterns and practices guidance in action. Microsoft patterns and practices reference implementations cover such topics as application integration and interoperability using Web services.
- **Application Blocks or Enterprise Library for .NET Framework.** Application blocks are reusable source-code components that provide proven solutions to common development challenges. They can be integrated as is into applications, or they can be extended or customized. Microsoft patterns and practices application blocks address specific recurring

problem domains such as data access, logging, user interface process, cryptography, security, configuration, exception handling, and composite user interfaces.

Note More information on Microsoft patterns & practices is available at http://msdn.microsoft.com/practices/.

High-Performance and Distributed Computing

The techniques and technologies for high-performance distributed computing can affect a migration. High-performance computing can change the underlying system architecture because of the requirements for computing power and network bandwidth. In addition, the potential distributed nature of high-performance computing raises issues with the test and deployment teams, both of which now need to consider multiple computers instead of single computers. With all the power now available on desktops and servers, the scientific and engineering community looks for ways to maximize the processing power of this commodity hardware and, potentially, the used cycles of these systems.

Two possible approaches are worthy of consideration:

- **High-performance cluster computing.** This refers to a technique where large numbers of workstations or servers (or a combination of the two) work together to perform jobs that, until recently, only supercomputers could perform.
- **Grid computing.** This refers to using the network to link multiple computers so that tasks can be shared between them. Again, programmers and users can access significant computing power through the use of grid technologies. Furthermore, good use can be made of such CPU cycles, whereas before they would go to waste.

Note More information on high-performance computing is available at http://www.microsoft.com/technet/itsolutions/cits/interopmigration/unix/hpcunxwn/default.mspx and http://www.microsoft.com/windowsserver2003/ccs/default.mspx.

Further Reading

You can also refer to the following Microsoft Press® books for more information on specific technology.

Reference Architectures

Microsoft Systems Architecture—Enterprise Data Center

Microsoft Systems Architecture—Internet Data Center

Application Architecture for .NET: Designing Applications and Services

Microsoft SQL Server 2000 High Availability Series: Volume 1: Planning

Microsoft SQL Server 2000 High Availability Series: Volume 2: Deployment

Managing and Maintaining a Microsoft® Windows Server™ 2003 Environment

Managing Projects with Microsoft® Visual Studio® 2005 Team System

Microsoft® Systems Management Server 2003 Administrator's Companion

Microsoft Active Directory Branch Office Guide: Volume 1: Planning

Microsoft Active Directory Branch Office Series Volume 2: Deployment and Operations

Reference Building Blocks

Data Access Application Block for .NET

.NET Data Access Architecture Guide

Designing Data Tier Components and Passing Data Through Tiers

Exception Management Application Block for .NET

Exception Management in .NET

Monitoring in .NET Distributed Application Design

Microsoft .NET/COM Migration and Interoperability

Production Debugging for .NET-Connected Applications

Authentication in ASP.NET: .NET Security Guidance

Building Secure ASP.NET Applications: Authentication, Authorization, and

Secure Communication

Development References

Inside Microsoft® SQL Server™ 2005: T-SQL Programming. Itzik Ben-Gan. (Solid Quality Learning) and others. 0-7356-2197-7

Inside Microsoft® SQL Server™ 2005: T-SQL Querying. Itzik Ben-Gan. (Solid Quality Learning) and others. 0-7356-2313-9

Inside Microsoft® SQL Server™ 2005: The Storage Engine. Kalen Delaney. (Solid Quality Learning) 0-7356-2105-5

Inside Microsoft® Visual Studio® .NET 2003

Programming with Managed Extensions for Microsoft® Visual C++® .NET

Programming with Managed Extensions for Microsoft® Visual C++® .NET Version 2003

Index

A

B

C

D

E

F

G

H

I

M

N

O

P

R

S

T

W

X