



realtimepublishers.com[®]

The Definitive Guide[™] To

Enterprise Change Management



Dan Sullivan

Chapter 4: Managing Change in System Configurations	68
Interdependence in System Configurations	68
Understanding Implementation Details	69
Details Reveal Dependencies.....	70
Hardware Dependencies	70
Software Dependencies.....	71
Rippling Effects of Misconfigured Systems.....	72
Defining the Goals of System Configuration Management.....	73
Accounting for System Services.....	74
Charging for Services	74
Software License Compliance	75
Planning Upgrades and Retiring Equipment.....	75
Redeploying Assets.....	75
Managing Leases, Warranties, and Other Contracts.....	76
Monitoring System Performance	76
Measuring System Load	77
Measuring Resource Usage.....	78
Maintaining a Secure Infrastructure.....	78
Keeping Systems Updated with Patches.....	79
Monitoring Suspicious Activity	80
Logging Vulnerabilities, Breaches, and Responses.....	81
Fault Management	81
Identifying Failed Components.....	82
Isolating the Effects of Failures	83
Correcting Failure	84
Managing Configurations	84
Modeling System Configuration Management.....	85
Defining System Configuration Assets.....	86
Identifying Dependencies Between Assets.....	86
Developing, Publishing, and Enforcing Policies	87
Roles and Workflows in System Configuration Management	87
Understanding the Unique Challenges of System Configuration Management	88
Summary	88

Copyright Statement

© 2003 Realtimedpublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimedpublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimedpublishers.com, Inc or its web site sponsors. In no event shall Realtimedpublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtimedpublishers.com and the Realtimedpublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimedpublishers.com, please contact us via e-mail at info@realtimedpublishers.com.

Chapter 4: Managing Change in System Configurations

IT infrastructure is a lot like an orchestra—they comprise a wide range of instruments that act in concert to create a unified product, be it music or information flow. When the instruments work together, the results are favorable; however, a slight variation in even one instrument—a wrong note from the French horn or a misconfigured router table—can skew the overall operation of the group. Managing system configurations is like conducting an orchestra—it takes knowledge of each instrument’s role in the overall process, the ability to identify variations from the ideal, and a clear vision of the final product of your efforts.

This chapter focuses on managing the hardware, software, and network components that are the foundation of an IT infrastructure. We’ll explore problematic system configurations that have consequences that affect more than just network services. I’ll provide examples that demonstrate the subtle dependencies between system configurations and business operations. Next, the chapter outlines the goals of system configuration management followed by a discussion of configuration management and the ECM model outlined in Chapter 2. Finally, I’ll briefly discuss challenges particular to system configuration management.

Interdependence in System Configurations

An IT infrastructure is made up of a wide range of devices that perform a variety of functions and create many interdependencies. The effects of such interdependencies can be obvious—large-scale failures or misconfigurations such as an offline database server or an overtaxed email server—or less easily identified. It is much more difficult to identify problems with smaller or embedded components although the results in many instances are just as detrimental as large-scale problems.

For example, consider a business intelligence application that uses a moderately sized data warehouse. Executives, LOB managers, and project managers depend on this data warehouse for reporting on key performance indicators, operational measures, and ad hoc querying. Users access the data warehouse through a portal and a Web-based interface to the database (see Figure 4.1). As long as the portal server, the database server, and the network are operating, the system is functioning, right? Not necessarily.

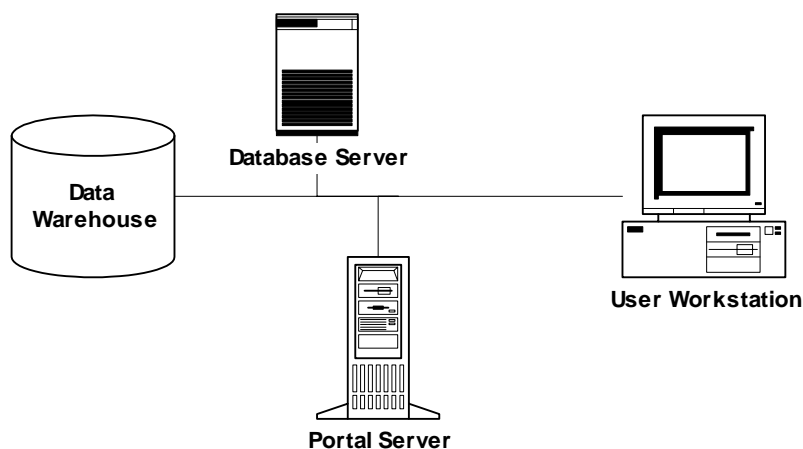



Figure 4.1: A small business intelligence system looks simple at a high level, but this figure hides the complexity of the underlying infrastructure.

Understanding Implementation Details

Let's take a closer look at the components. The device labeled data warehouse stores the information managed by the database server. In more detailed terms, the data warehouse is a storage area network (SAN)—an array of disks configured to act as a single logical storage device.

 SANs have many advantages for network administrators. They allow multiple servers to share disk storage, reducing the chance of wasted space on individual servers. A single SAN requires less administration than several separate server-based storage systems.

How SANs Fit in the Infrastructure

In a typical SAN environment, multiple servers will use a single SAN for storage, as Figure 4.2 shows.

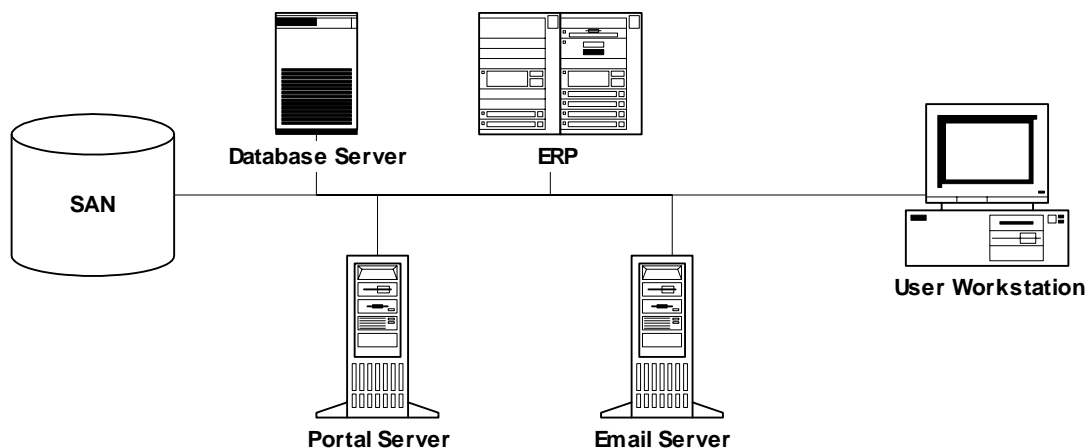


Figure 4.2: Multiple servers often depend upon a single SAN.

Data moves in and out of the SAN through channels that respond to requests for data from controllers found on servers. Channels are like traffic cops, they decide which data moves and how the data moves. When a server makes a request for data, the channel determines how much bandwidth to allocate to that server based on the overall demand on the SAN. For example, if three servers are pulling data from the SAN, the controller might allocate 25 percent of the controller's bandwidth to the fourth server. If only a single server is retrieving data at one time, the controller might allocate 50 percent of the controller's bandwidth and keep 50 percent in reserve in case another server makes a request. If, after some time, no other server makes a request, the controller may decide to allocate 100 percent of the bandwidth to the one active process.

Typically, SANs have multiple channels (see Figure 4.3). Each channel manages a set of disks in the array. This setup allows SANs to handle multiple requests for data more efficiently. Rather than force all data to move through a single controller, the SAN can move data through several controllers in parallel.

To optimize data flow, administrators split data that needs to be accessed at nearly the same time across different controllers. For example, data warehouses make extensive use of indexes and tables of data. Indexes can quickly pinpoint the locations of specific pieces of data, such as the number of products sold in the Northeast region in the fourth quarter of 2003. That location information is then used to actually find the data on the disk and retrieve it. To keep index operations from contending with data-retrieval operations on the same controller, the indexes and data files are kept on separate controllers.

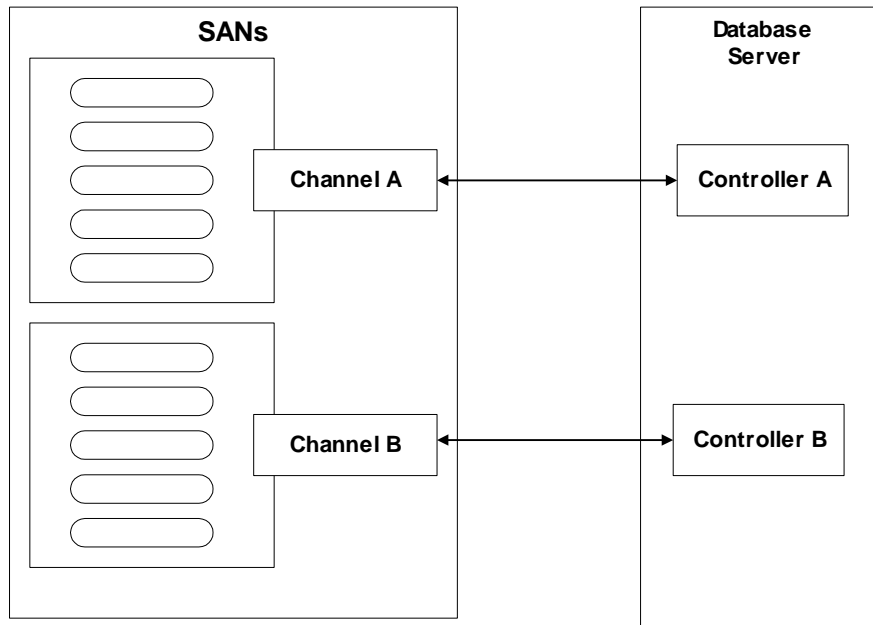



Figure 4.3: Multiple controllers manage data moving to and from disks.

Clearly, parallel operation in the SAN can improve I/O performance just as parallel CPUs on a server improve compute-intensive operations. However, this benefit requires dependencies between infrastructure components such as the SAN and servers.

Details Reveal Dependencies

Digging into the implementation details of network devices often reveals dependencies that are easily missed when working at the higher level of servers and SANs. In the following sections, we'll explore hardware and software dependencies.

 Thanks to Robert Webb of Venturi Partners for providing these examples.

Hardware Dependencies

As Figure 4.3 shows, the SAN channels and the database servers are tightly coupled. Data moves directly from one to the other, which implies that the two must have compatible modes for transferring data, particularly the speed at which data is transferred. The maximum transfer speed of the slower of the two will be the maximum transfer speed of the system. A 250Mbps SAN channel will have an effective throughput of only 40Mbps if the servers have 40Mbps controllers.

High-speed network devices and I/O equipment are limited by the speed of other devices with which they communicate. Changes in IT infrastructure should be assessed at the system level, not just the level of individual components, to identify dependencies that will limit performance of equipment.

Software Dependencies

Sometimes the dependencies between elements of networked systems are less obvious. Consider the relational database management system (RDBMS) for a data warehouse. Many RDBMS vendors offer parallel processing functions to improve performance. Parallel query processing features take advantage of multiple CPUs on a server to divide a user's query into multiple tasks that execute concurrently on different CPUs. Parallel partitions split data in a table into multiple chunks called partitions. From a user's or developer's point of view, all the data is in a single table; the RDBMS is able to spread the data over multiple disks by putting each partition on a separate disk (see Figure 4.4).

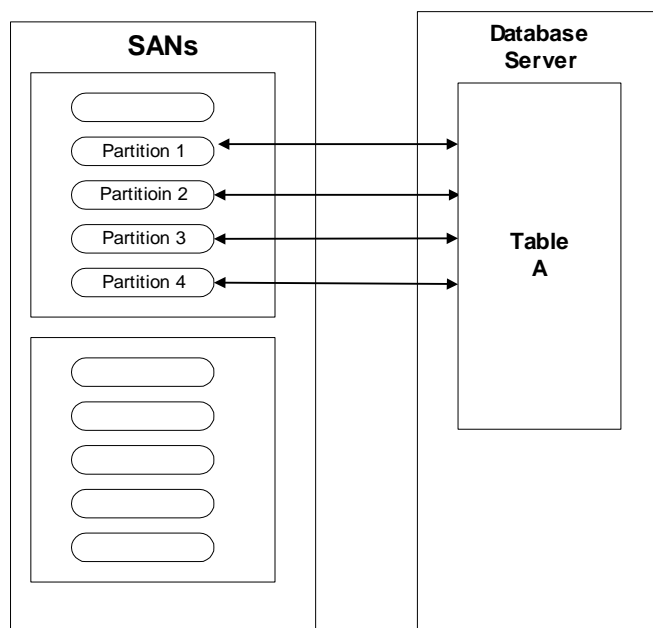


Figure 4.4: Partitioning is a software-based technique that takes advantage of parallel disk operation to reduce the time required to respond to queries and manipulate data.

In general, the RDBMS with parallel processing features will break down a task in such a way as to have one subtask for each CPU. A database server with four CPUs will break a task into four separate processes, and each process will execute on a separate CPU.

Again, problems can occur when the elements of the overall system are not configured properly. If the CPUs are too slow to process data-intensive queries, CPU utilization will reach 100 percent. Because the task has been parallelized to four subtasks, all four CPUs could reach 100 percent utilization for a single query. This behavior effectively blocks other users from executing queries or data load operations.

When hardware is not properly configured, software features designed to improve performance can introduce unexpected performance problems.

Rippling Effects of Misconfigured Systems

Misconfigured systems can result in suboptimal performance, poor user adoption of applications, costly changes to other parts of a system to improve performance, and, ironically, the purchase of additional hardware to achieve expected throughput. (Of course, additional hardware just compounds the problem by adding yet another element that needs to be properly configured.)

For example, consider a high-speed SAN coupled with a server using slow controllers. The decision to purchase that major piece of storage equipment was probably driven, in part, by the need to improve I/O performance on the data warehouse. System designers need to consider the controllers on the database server. Are high-speed controllers available for the server? The controllers need to move data along the internal data bus on the server as well as to and from the SAN. If the internal data bus cannot support high-speed controllers, the server needs upgrading. Some database vendors license RDBMS software based on CPU performance, so upgrading the server could entail simply upgrading a software license. The cost of solving an I/O performance problem can quickly move from the realm of replacing a single storage device to performing major hardware and software upgrades (see Figure 4.5).

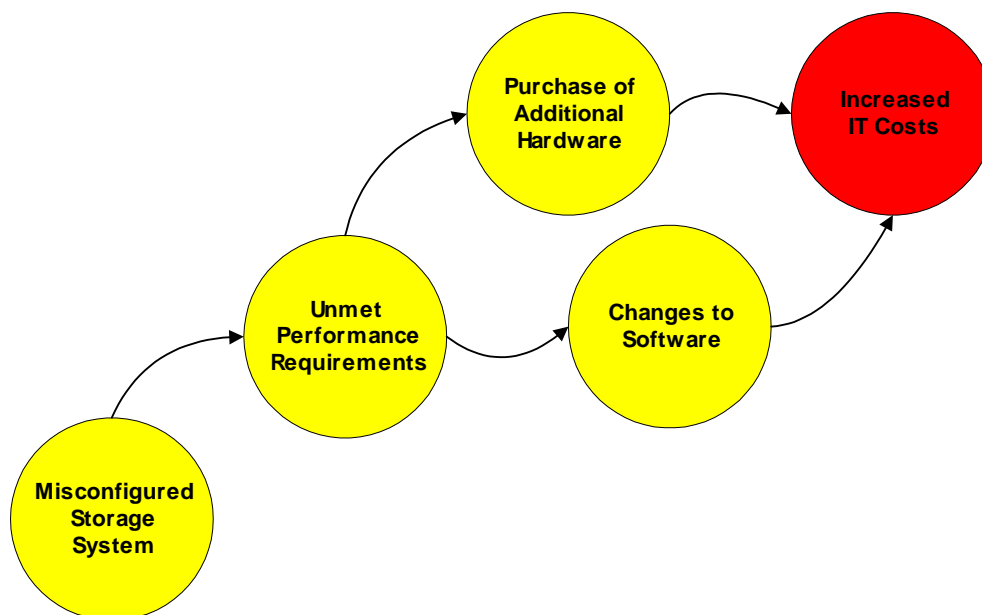



Figure 4.5: *The ripple effect of misconfigured systems often includes unmet requirements and attempts to fix the symptoms without understanding the underlying problem.*

 CPU-based licensing of software can significantly increase the cost of hardware upgrades. Be sure to include these costs when evaluating system upgrades.

In the database partitioning example, the impact of improper configuration can reach end users. As noted, partitioning features of relational databases create processes that can monopolize CPUs and delay the execution of other users' operations. Business intelligence systems are designed for rapid response to ad hoc queries—even large, data-intensive queries—so anything that slows response time undermines system design. Users expect responsive reporting from business intelligence systems, and poor performance will send executives, managers, and analysts to other sources for their information (see Figure 4.6).

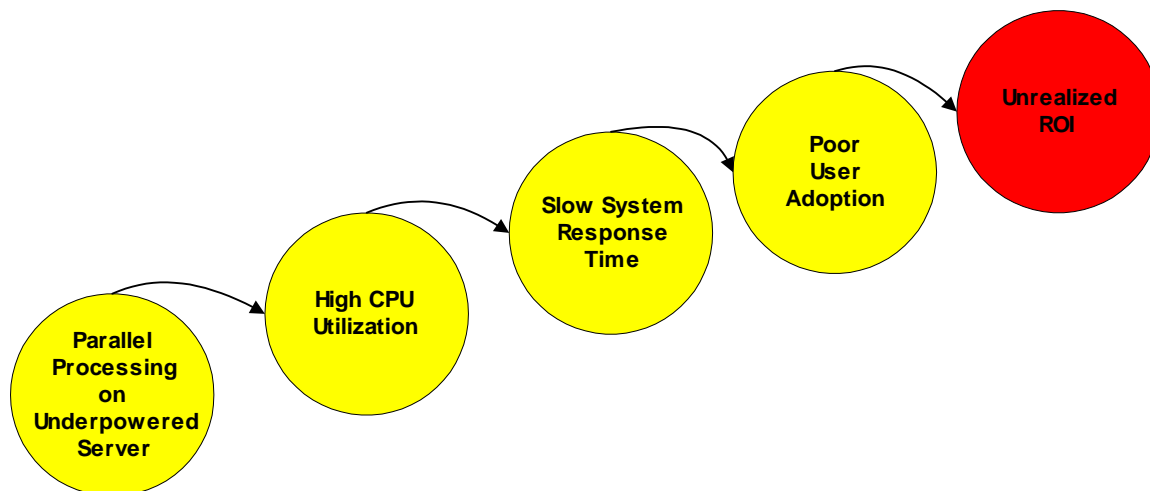



Figure 4.6: Misconfigured systems can undermine user adoption and limit the ROI realized by a system.

Clearly, the ripple effects of change can extend far beyond the server room and affect business operations. Fortunately, ECM techniques provide the means to manage change in this highly distributed and interconnected environment. Before turning to discuss how you apply the ECM model to system configuration management, let's outline the primary goals of system configuration management.


 See Chapter 2 for a detailed discussion of the ECM model.

Defining the Goals of System Configuration Management

System configuration management addresses both business and technical goals. The five main categories include:

- Accounting
- Performance monitoring
- Security
- Fault management
- Configuration management

Together these categories address the main problems involved with keeping an IT infrastructure operating.

 The International Telecommunication Union (ITU) has worked to define the characteristics of these five functions. You can find more information about these topics at <http://www.itu.int/home/> (membership is required to access some information).

Accounting for System Services

Accounting in the area of system configuration does not duplicate the efforts of a finance department; it complements them. As IT departments become responsible for charging back costs to customer departments, it is essential that they track who is using resources and how much those resources cost. Accounting for system services also supports internal IT operations such as:


- Software license compliance
- Upgrade/end of life planning
- Redeployment
- Lease, warranty, and other contract management

It is a costly oversight to simply not perform accounting for infrastructure assets. According to the Gartner Group, organizations that implement asset management systems save as much as 30 percent per asset in the first year and an additional 5 percent to 10 percent in annual costs for the next 5 years per asset (Source: “Enterprise Management ROI and Cost Reduction in 2003”—*Gartner*, November 2002—by M. Nicolle, K. Brittain, and P. Adams).

Charging for Services

Charge backs to customers present a number of challenges. First, much IT infrastructure is a “common good.” Like roads, parks, and police services, the benefits are shared across the community and it is impossible and impractical to measure individual’s use of each service. Costs are allocated according to some measurable use, such as the number of users in a department or the IT budget for other services.

Second, IT departments procure equipment and services from multiple vendors. Sometimes equipment and services are purchased; other times they are leased. Some carry warranties; others do not. Some software licensing fees are based on CPU performance; others are based on named users. Some items carry maintenance agreements; others do not. The wide variety of vendor arrangements requires the implementation of workflow processes and document management practices to manage change.

 Chapter 5 discusses document management and its relation to ECM.

Third, IT services and users change constantly. New applications are brought online. Legacy systems are integrated with new e-commerce applications. Servers that once ran a single application are consolidated into a single server. Segments are added to the network. New services, such as spam filtering for email, are added. The user base changes as well.

Departments cut back on staff and reduce the number of ERP users. Marketing departments implement strategic plans and increase their use of business intelligence systems. To adequately account for the dynamics of IT use, accounting systems need to allocate charges on a short-term basis (for example, monthly). Estimating costs annually during budget development will not accurately reflect the use of IT services.

Software License Compliance

To remain in compliance with software licenses, IT needs to know which software is installed on servers and PCs and, in some cases, how many users are using the software. Automated tools can help inventory PC-based applications. In addition to providing accurate accounts of standard programs, these applications can identify non-standard software that has been installed on PCs without approval. Maintaining compliance on server software can be more difficult.

When software is licensed by CPU, it is important to track any server upgrades or downgrades. Replacing a single CPU server with a 4-processor server can knock you out of compliance. However, redeploying a 4-processor server and replacing it with a 2-processor server would not necessarily put you out of compliance—but could leave you paying more than you should be.

When server software is licensed by named users, administrators should have ready access to changes in a user's status. Enterprise directories and single sign-on servers can provide information about new and terminated employees and contractors. (Single sign-on systems allow administrators to grant or revoke access to multiple applications from one point, so reports on changes by application can simplify ensuring license compliance). When these systems are not in place, administrators typically depend on manual methods to track changes.

Planning Upgrades and Retiring Equipment


Accounting systems provide the raw data needed to plan upgrades and retire equipment. With a database of equipment specifications, IT can project the useful lifetime of an element in the infrastructure and estimate the baseline cost of maintaining the current level of service over a number of years. When combined with utilization information, the accounting database can help with redeployment decisions.


Redeploying Assets

There are several reasons to redeploy assets, including:

- **Eliminating underutilization**—Many application servers are underutilized. During the late 1990s, the conventional wisdom was to install a new server for every application, such as Web servers, email systems, databases, and content-management systems. In some cases, this setup eased software administration by eliminating conflicts between applications running on the same machine and minimizing the number of applications affected by maintenance or a server failure. The downside was that this practice led to a proliferation of servers with excess capacity. It also increased the number of servers that had to be locked down and patched to maintain a secure environment.
- **Reducing licensing fees**—Many organizations are now consolidating servers and running multiple applications on a single server. Doing so can reduce OS licensing costs. Although the total number of users or CPUs on a consolidated server may be greater than on individual servers, the marginal cost of adding those users or CPUs to an already licensed server is less than the total cost of individual licenses for multiple servers.

- Changing OS platforms—In some cases, it is not the hardware but the software that is redeployed. Concerns about security and license costs are leading some organizations to consider deploying applications on the open source platform Linux. Although many Linux distributions are free, enterprise-scale versions of Linux are not. Licensing is only one part of the cost of ownership equation, and, to date, there is no consensus on the relative cost of Linux versus proprietary OSs.

 Many vendors offer their perspective on Linux and its role in the enterprise. For more information, see “Total Cost of Ownership of Linux in the Enterprise” at <http://www-1.ibm.com/linux/RFG-LinuxTCO-vFINAL-Jul2002.pdf> and “A Closer Look at Linux” at <http://www.sun.com/2002-0319/feature/>.

 There is still debate about the total cost of ownership (TCO) of Linux versus other OSs. For a couple of perspectives, see “Linux TCO: Less Than Half the Cost of Windows” at http://www.ciupdate.com/article.php/10493_1477911 and “Linux Servers: No ‘Silver Bullet’ for Total Cost of Ownership” at <http://www.microsoft.com/windows2000/docs/od1035.pdf>.

Managing Leases, Warranties, and Other Contracts

The “do more with less” edict that has come down to many IT departments is driving administrators and managers to save wherever possible. One way to prevent unnecessary expense is by managing leases, warranties, and other contracts effectively. For example, leases may differ on which party is responsible for maintenance and repair of equipment. Having up-to-date access to lease details for all leased equipment is essential to avoid covering the cost of services for which the organization is not responsible. Similarly, exercising warranties can help minimize service and repair costs.

Accounting for enterprise assets is not the core function of IT and, frankly, many administrators would rather have nothing to do with it. Nonetheless, to support change management with respect to IT infrastructure, we need asset accounting. Ideal ECM systems offer functions for allocating costs, supporting upgrading and redeployment decisions, determining end of life and replacement cycles, and managing leases, warranties, and contracts.

Monitoring System Performance

Complex IT systems take much effort to design, install, and configure. It would be nice if after all that work they could run by themselves, but, marketing slogans aside, it just doesn’t happen. These systems need regular monitoring. In particular, administrators need to monitor systems for:

- System load
- Resource usage
- Security breaches

Measuring System Load

A simple phrase like system load can mean many things. When monitoring a server, administrators will check the number of processes executing, the number processes waiting for resources, and the time required to respond to a request, such as to display a Web page or retrieve a result from a database query. When measuring the load on a network, administrators monitor bandwidth utilization (how much data is moving through the network or a segment of the network) and latency (how long it takes to move a packet of data from one point in the network to another). To monitor the load on a custom application, systems administrators require custom key performance indicators. For example, an ERP may be monitored for the number of users, the number of transactions processed, or the time required to execute particular tasks, such as generating a report. Monitoring applications, such as the Windows Performance Monitor that Figure 4.7 shows, capture and display information about a variety of key indicators.

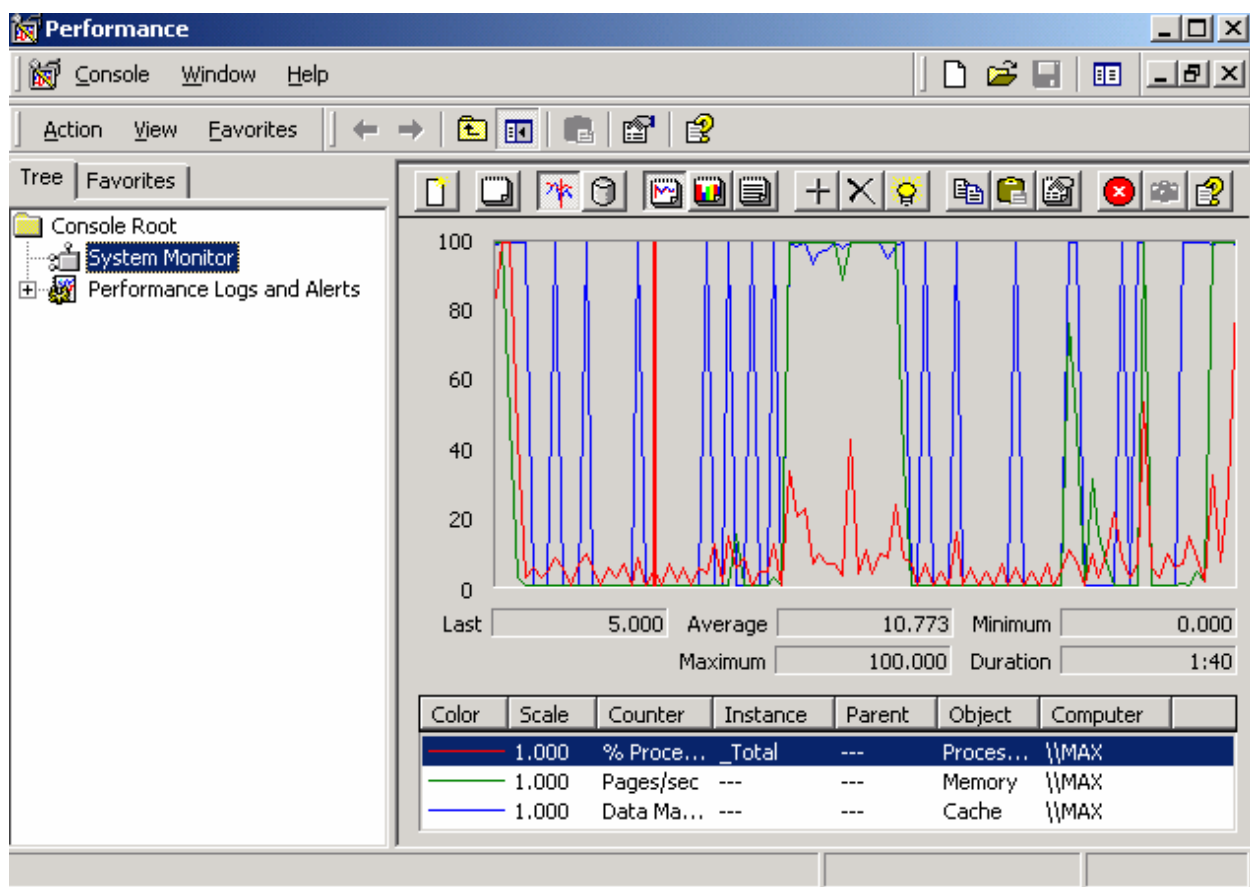


Figure 4.7: OS performance monitoring programs track an array of measures for monitoring processors, memory, I/O, network operations, and sometimes applications.

A common aspect of performance monitoring, regardless of the type of system, is the need for baseline measures, which allow administrators to track trends, identify periods of unusually low or high utilization, and, to some degree, forecast loads. From an enterprise change perspective, administrators need access to baseline information as well as measurements over time that require monitoring processes to be running and logging information. These processes are ideally managed in an ECM system.

Measuring Resource Usage

Administrators can gain a quick picture of overall system load by checking a few performance statistics, such as CPU utilization, free disk space, and total bytes per second transmitted and received on the network. These indicators give an overall assessment of system performance, but may not be sufficient to identify problems.

For example, if CPU utilization is unusually low, the cause could be one of several factors, including fewer users than normal. Not all problems are that easy to identify. The problem may lie with another resource on the system. A slow disk can cause processes to wait while data is retrieved. I/O-bound processes such as these do not use the CPU while waiting. Similarly, a low-bandwidth network can keep processes waiting while data is sent to and retrieved from other servers.

Hardware systems are like production lines. The overall operation is limited by the speed of the slowest step in the process. To identify bottlenecks, administrators need information about the performance of particular resources:

- CPUs
- Disks
- Network interfaces
- Network hubs
- Routers
- Memory
- Caches

Within an ECM framework, administrators will need access to both high-level system load performance statistics and fine-grained resource-level statistics.

Maintaining a Secure Infrastructure

Maintaining a secure infrastructure is one of the most challenging aspects of system management. Accounting and performance monitoring are well-defined operations and are largely in the control of IT staff. Security deals with “what might happen” and the potential list of problems is long. Hackers might:

- View or copy confidential information
- Destroy data
- Use resources for malicious purposes
- Masquerade as a legitimate user
- Install viruses, worms, Trojan Horses, and other destructive programs

Although enterprise security is well beyond the scope of this book, ECM practices are an important part of an overall security practice. Changes in hardware and software can introduce vulnerabilities in a number of ways, three of the most common vulnerabilities are the result of default configurations, software bugs, and unnecessary services:

- **Default configurations**—Software applications sometimes have default configurations that should be changed upon installation. The VAX/VMS OS, for example, used to ship with three privileged accounts set with default passwords. Past versions of Oracle installed with a privileged account and a default password. In both cases, someone with a little administrative experience with either application could gain access to other systems.
- **Software bugs**—Installing or upgrading an application can introduce vulnerabilities to an organization. A common hacking technique is to generate a “buffer overflow” in an application. Doing so basically entails sending more data to an application than it expects and causing data already in the application to be overwritten. Sometimes this attack only destroys the overwritten data. In other cases, hackers replace data with instructions to perform an operation that ultimately leads to more severe damage.

In the spring of 2003, a bug in Microsoft SQL Server allowed hackers to gain sufficient control of servers to have the systems flood the Internet with bogus traffic and effectively shut down Internet services for some organizations.

- **Unnecessary services**—OSs, including Microsoft Windows Server, Linux, and UNIX, install commonly needed services by default. Although this configuration saves administrators time during installation, it creates potential points of entry for hackers. Bugs have been found in basic services. If services are not needed, they should not be installed.

The solutions to these three problems are basic:

- Change default passwords
- Install corrected versions of the program or parts of the program known as patches
- Install only services needed by a server

In all cases, administrators must log the changes and applied patches and services in a change-management system. When a vulnerability such as the SQL Server bug becomes known, organizations with complex infrastructures will need to quickly identify and correct potentially affected systems.

Keeping Systems Updated with Patches

Keeping track of patches can be an administrative nightmare. First, one needs to track all the software in an organization, including everything from email clients on users’ desktops to the ERP system that is the backbone of day-to-day operations. For each piece of software, administrators must know the precise component and version number (not just Oracle 9, but Oracle Net 9.2.0.1.0). Patches vary by OS platform, so this information must also be tracked. Of course, the patches themselves must be tracked too.

In addition, administrators must stay updated on new releases from vendors, or, in the case of open-source software, from development teams. When patches are released, administrators must understand how the patch will affect the application to which it applies as well as the systems that are dependent upon that application. Although vendors do their best to correct problems rather than introduce them through patches, occasionally a fix breaks working systems (see Figure 4.8).

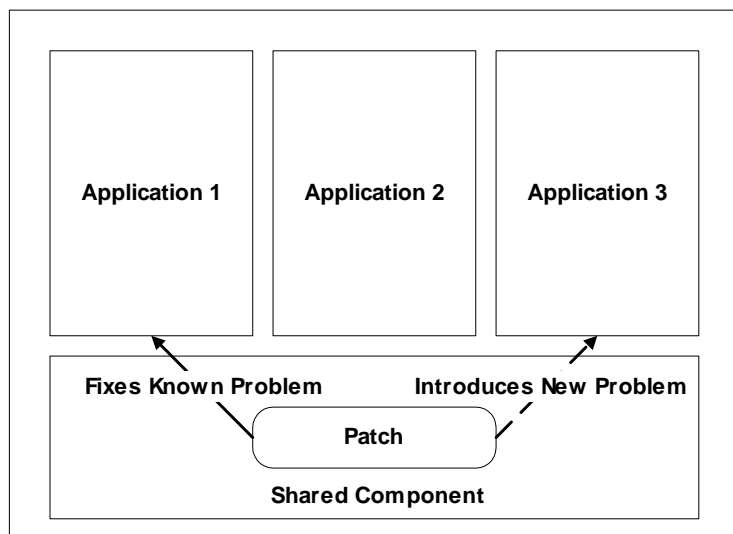


Figure 4.8: Patching a component used by multiple applications risks fixing one problem at the expense of introducing another.

An ECM system allows administrators to track software details as well as the results of applying patches. A simple note in the ECM repository can save administrators hours of installing and uninstalling a patch that does not function correctly in their environment.

Monitoring Suspicious Activity

ECM systems support security monitoring activities. Specialized applications, such as intrusion detection systems and file change-monitoring systems, generate a great deal of information in log files. In most cases, administrators purge log files on a regular schedule, but some should be kept in the change-management system, including:

- Baseline logs when security applications are first executed
- File change detection logs generated after an application is installed on a server
- Intrusion detection logs generated during a security breach

Entries in the ECM system should also include narratives that describe the reason for saving the log and responses to suspicious activity.

Logging Vulnerabilities, Breaches, and Responses

Many organizations are understandably reluctant to discuss security breaches, but that should not stop administrators from sharing details of vulnerabilities, breaches, and responses. ECM systems should be used to record these details and provide a single point of access to that information for internal staff. This record is important in large geographically distributed organizations that employ many systems administrators. When an administrator finds or corrects a problem with mission-critical software, such as server OSs and email servers, the administrator can save others time and duplicated effort by recording and making available this information.

Security depends on knowledge of your infrastructure and its vulnerabilities. ECM systems should track much of the information administrators need to prevent and respond to security breaches.

Fault Management

Fault management deals with identifying and correcting problems in the IT infrastructure. The starting point for correcting problems is to understand exactly how the system *should be* functioning. For example, traffic between any two segments of a three-segment network should be routed directly between the segments. In the case of failure in the network between two segments, the traffic is rerouted through the third segment as Figure 4.9 shows. The problem may not be immediately apparent because of redundant paths through the network. Recognizing there is a problem is the first stage of fault management.

The basic steps of fault management are:

- Identifying a failure in the system
- Isolating the effects of the failure
- Correcting the failure

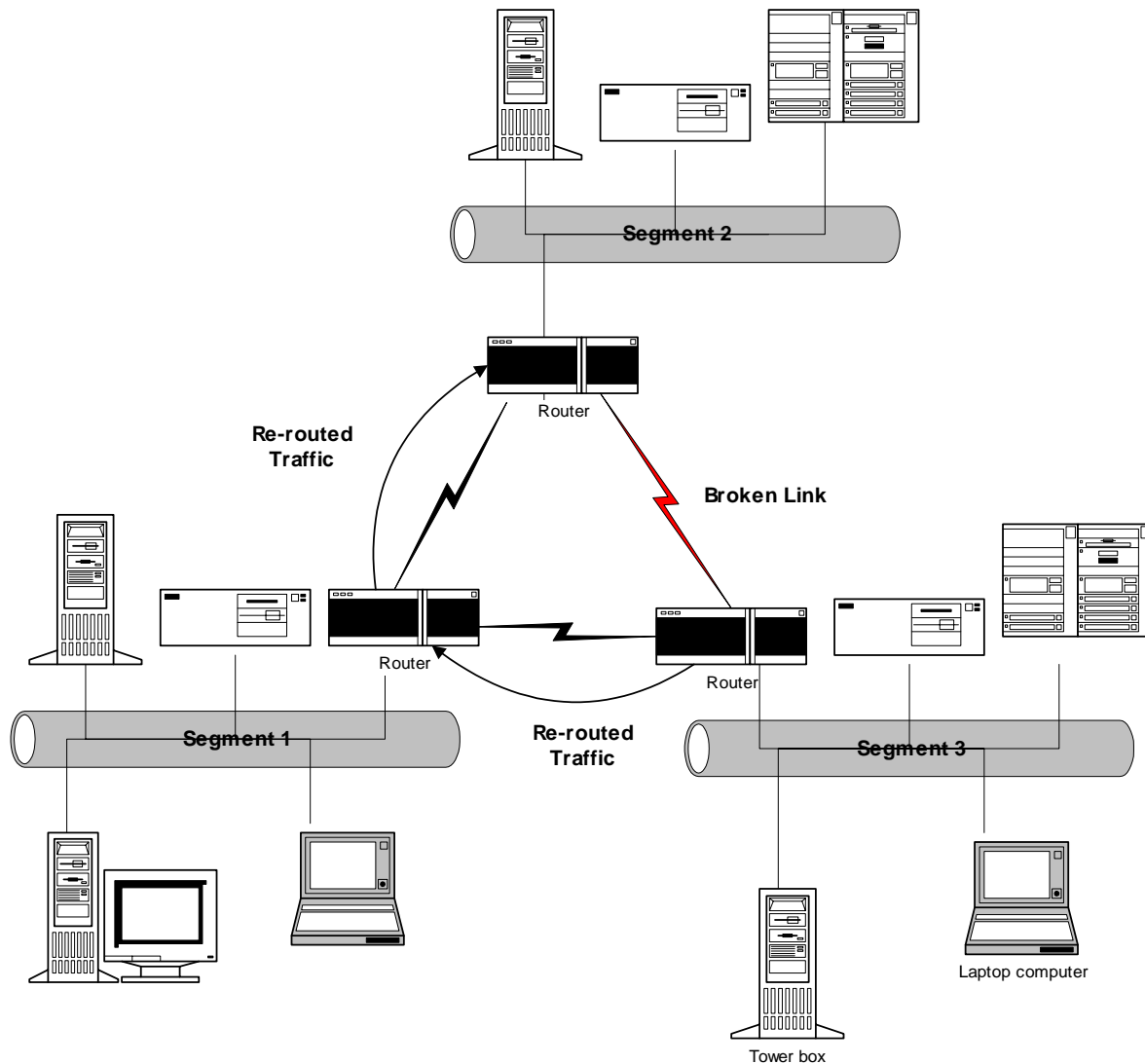


Figure 4.9: Network traffic continues to flow between network segments even during a failure.

Identifying Failed Components

When a major component, such as a server or router, has failed, it is fairly easy to identify. Other failures take more investigation into dependencies between components. Consider an e-commerce system that seems to accept a customer's order but returns a standard browser error message such as *This page cannot be displayed*. The problem could be:

- The Web server is down and cannot display a confirmation page for the customer.
- The database server is down and sales transactions are not recorded. The interface component of the e-commerce system, which is waiting for the database server to return, does not respond to the Web server, which, in turn, generates the error message.
- The database server is not down but is out of temporary work space and cannot process any more transactions.

- A network administrator closed a network port on the application server during a routine security review not realizing that developers had recently moved the e-commerce application to that port.
- A Denial of Service (DoS) attack is underway against your site and the network is flooded with bogus traffic.

There are many other possible causes. Problems with networks and distributed systems often show themselves through symptoms far removed from the root cause. For these situations, systems administrators and network engineers need information about assets that make up systems such as an e-commerce or ERP application, and the dependencies between those assets. With this information in hand, they can devise tests to identify the cause of the problem.

Isolating the Effects of Failures

Once the problem component or system has been identified, it may require isolation. In some cases, such as a failed printer, you do not need to isolate the device, although you might shut down the print queue to keep jobs from building up in the queue.

When an essential component fails, backups must be used. Critical servers are sometimes run in clusters sharing the workload between them. If one server fails, the others automatically pick up the work of the failed server. Networks are often designed with redundant paths (as Figure 4.9 illustrates). Again, this design provides for automatically adapting to the failure.

Other failures require manual intervention. For example, business intelligence servers generally do not require automatic failover protection. If a report server is down for a short time, it won't affect short-term business operations. If the failure is prolonged, a backup server may need to be manually configured and put online. In large organizations, an ECM system could provide information about similarly configured servers, their core functions, typical utilization, and other information that can help to identify a substitute server.

When failures are caused by viruses and worms, information in the ECM system can help to isolate the problem. Take the case of worms—programs that copy themselves from one system to another. One way they can spread is through trust relationships between servers. The idea behind a trust relationship is that an OS on server A will allow an administrator account or process from server B to access the resources of server A as if the account were an administrator account on server A. This relationship eases some systems administration tasks but creates a serious problem when one server in the trust relationship is compromised.

Again, information kept about system configurations in an ECM system could be used to identify servers trusted by a compromised server and list the services running on each server. Administrators could then identify which servers are vulnerable to worms that make use of bugs in particular services.

One of the challenges in isolating failures is that the affected servers may not be physically near each other. In the case of trust relationships, the servers could be anywhere on the network (see Figure 4.10).

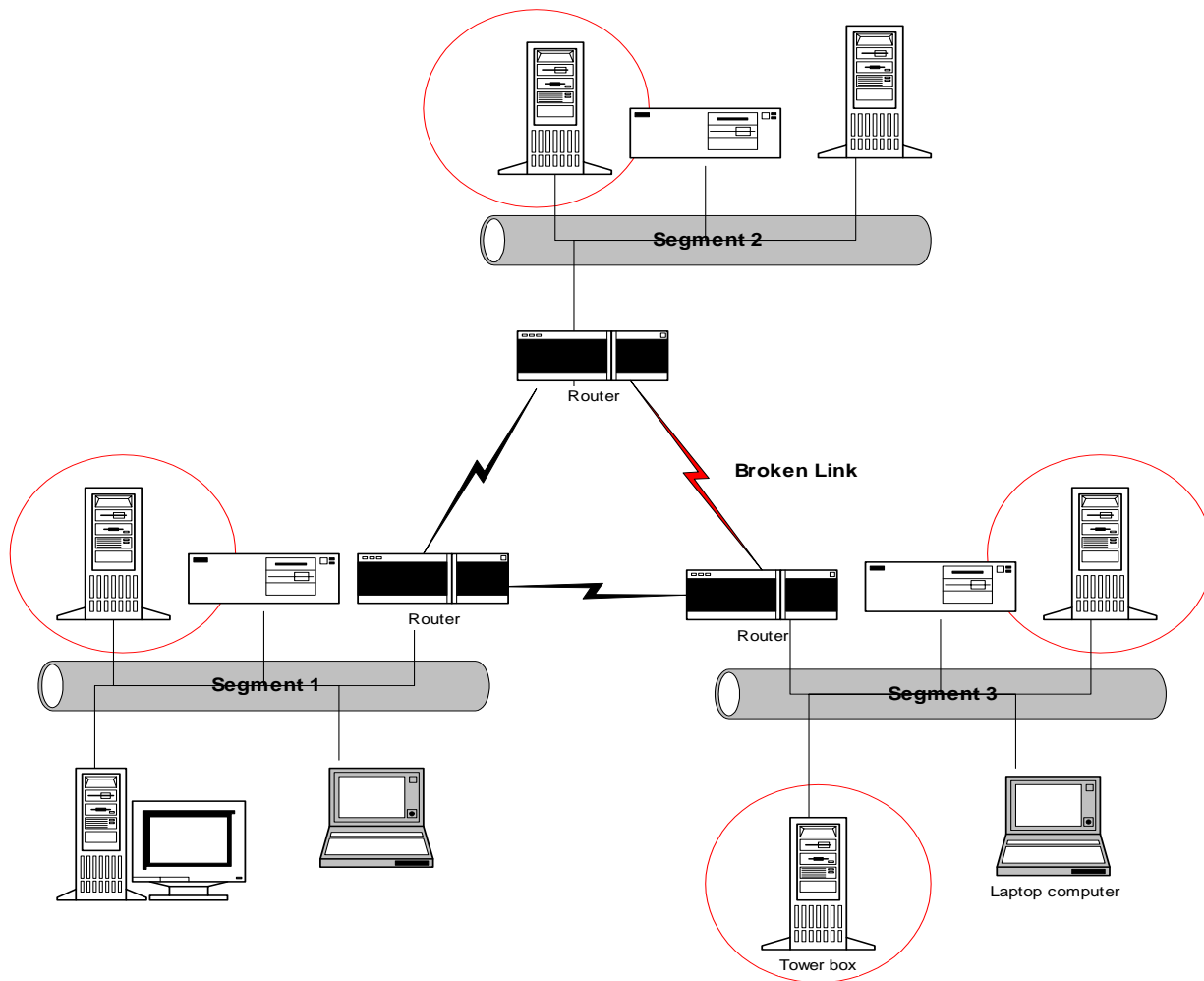


Figure 4.10: Isolating failures in distributed systems requires details about logical relationships, not just physical relationships, between servers. Servers that trust each other are encircled in red.

Correcting Failure

The solutions to failures are as varied as the causes. Information about failures—how they were isolated and corrected—should be logged in an ECM repository for future reference. The repository then becomes a source of basic configuration information as well as a reference on past troubleshooting efforts.

Managing Configurations

The final goal of system configuration management is to keep accurate and precise information about infrastructure assets. Basic characteristics that should be tracked include:


- Device type and logical name
- Physical location
- Version of software elements
- Software patches applied

- Hardware revisions
- History of failures
- Brief description of purpose
- Responsible party and contact information
- Dependencies on other assets

Depending on the type of asset, administrators may also include:

- Performance monitoring statistics
- Security information
- Maintenance history
- Date acquired
- Date placed in current function

The information kept on each device should be driven by administrative tasks—such as maintenance, recovery, and security—and should be standardized across the organization.

 In Chapter 2, we explored the details of the ECM model discussed in the next section.

Modeling System Configuration Management

System configuration management, like other facets of ECM, can be modeled with information about five essential components:

- Assets
- Dependencies
- Policies
- Roles
- Workflows

System configurations are often modeled as collections of elements, networks, and services, so we will discuss that perspective as well.

Defining System Configuration Assets

IT infrastructure assets vary widely in function. Some of the more common network assets are:

- Servers
- Routers
- Bridges
- Switches
- Firewalls
- Wireless access points (WAPs)

These assets support services for the network, such as:

- Dynamic Host Configuration Protocol (DHCP) servers
- Domain Name System (DNS)
- Virtual private networks (VPNs)
- Email
- Instant messaging
- Voice over IP (VoIP)

These assets, while each providing a specific service, have multiple dependencies.

Identifying Dependencies Between Assets

System configuration management should model several types of dependencies. Hardware dependencies entail a physical connection between assets. For example, a server is physically connected to a switch, which is connected to a router, which is connected to a fiber optic backbone, which connects to an uplink to an Internet service provider (ISP), and so on.

Software dependencies occur when applications require other programs to function. Portal servers sometimes require client browsers with Java Virtual Machines (JVMs). Databases are designed for particular OS platforms. Many applications depend on TCP/IP to move information around the network.

Logical dependencies are created by the configurations of systems. The trust relationships supported by some OSs is an example of a logical relationship. A server's membership in a security domain is another example. Clusters of servers are a third example of a logical dependency.

The web of dependencies in even a simple network is too complex to model in precise detail. Your specific needs will determine the information you track. Most of us do not need to know the acceptable voltage range of the power supply in a server. ECM should be driven by pragmatics, not a quest to capture every conceivable bit of information about an infrastructure.

Developing, Publishing, and Enforcing Policies

System configuration management depends on policies to ensure a consistent and manageable environment. Policies and guidelines are developed for:

- Determining user access to applications and network resources
- Configuring firewalls
- Performing security audits
- Selecting OSs
- Configuring Web servers, email servers, and other communication applications
- Carrying out backup and recovery procedures

Roles and Workflows in System Configuration Management

Roles and workflows are closely related. In system configuration management, roles can be roughly broken down into the following categories:

- Element administrators—Responsible for specific devices, such as the routers in the Boston office, the four IBM DB2/UDB databases throughout the company, or the Apache Web server in the local office; element administrators are responsible for the basic operations of these devices and focus on tasks such as configuration, performance monitoring, and maintenance.
- Server managers—Responsible for services such as email and business intelligence reporting; services often depend on configuring and managing a number of different devices, so the scope of a service manager's responsibility spans a wider range than that of element administrators.
- Network engineers—Responsible for the backbone infrastructure on which services depend; network engineers' focus is on the reliable and efficient movement of data.
- Security managers—Unlike the other roles, security managers are not responsible for a device or service. Their focus is on the overall integrity of the system. They work with element administrators, server managers, and network engineers to configure systems, monitor activity, and conduct security audits.

Typical workflows span these roles. Some of the more common tasks are:

- Provisioning user accounts
- Monitoring performance
- Accounting
- Conducting security audits
- Backing up and restoring data

Like software development and document management, system configuration management is an essential part of ECM. Many aspects of system configuration management are captured in the change-management model, but this domain does have some unique challenges.

Understanding the Unique Challenges of System Configuration Management

Managing IT infrastructure poses two challenges not found in software development or document management. First, in many small and mid-sized organizations, network engineers must make changes directly to the production environment without the ability to first try the change in a test environment. Large IT organizations can support a network lab, but even a lab often cannot exactly duplicate the production environment—thus the need for ECM. Network engineers and system designers need to model the affects of change so that they can identify potential problems before implementing a change to the production environment.

Second, system configuration managers must assume that outsiders, and too often insiders, will try to attack their systems. Viruses, worms, and DoS attacks can severely affect day-to-day operations. Again, tracking detailed information about assets and dependencies can help administrators prevent and recover from security breaches.

Summary

System configuration management is evolving. Dependencies between assets are extending further and deeper into organizations, and the business need for reliable network and computing services is increasing. Enterprise-level system configuration management serves multiple needs, including accounting, performance monitoring, security, fault management, and component configuration management. Meeting these needs requires well-defined policies developed and enforced by server managers, network engineers, and security managers. Breaking out of the silo view of change management and addressing the issues presented in this chapter at the enterprise level is the next logical step in the development of system configuration management.

In Chapter 5, we will continue our examination of domain-specific change-management issues with a discussion of enterprise document management considerations. We'll explore how the ECM model enables us to understand the assets, dependencies, policies, and roles involved in document management within the larger picture of ECM.