z/OS

DCE Planning

GC24-5913-00

z/OS

DCE Planning

GC24-5913-00

Note

Before using this information and the product it supports, be sure to read the general information under Appendix A, "Notices" on page 55.

First Edition (March 2001)

This edition, GC24-5913-00, applies to Version 1 Release 1 of z/OS DCE Base Services, z/OS DCE user Data Privacy (DES and CDMF), z/OS DCE User Data Privacy (CDMF) (program number 5694-A01), and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

IBM welcomes your comments. A form for reader's comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Information Development, Dept. G60
1701 North Street
Endicott, NY 13760-5553
United States of America

FAX (United States & Canada): 1+607+752-2327
FAX (Other Countries):
Your International Access Code +1+607+752-2327

IBMLink™ (United States customers only): GDLVME(PUBRCF)
Internet e-mail: pubrcf@vnet.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use the information in any way it believes appropriate without incurring any obligation to you.

The following statements are provided by the Open Software Foundation.

Contents

Figures

Tables

About This Book

This book will help you plan for the organization and installation of IBM® z/OS DCE and IBM z/OS DCE Application Support (Application Support). It discusses the benefits of distributed computing in general, and describes how to develop plans for a distributed system in a z/OS environment.

Who Should Use This Book

This book is intended for the system planners, analysts, and programmers who are responsible for developing and implementing an installation plan for z/OS DCE or IBM z/OS DCE Application Support. Familiarity with distributed applications, system management, and network technologies will help you in developing your plans.

How to Use This Book

This book starts with a general description of distributed computing and progresses through to z/OS DCE specific hardware and software prerequisites.

Chapter 1, "Introducing Distributed Computing," introduces general distributed computing concepts and the Open Software Foundation Distributed Computing Environment (OSF DCE). Read this section if you are unfamiliar with distributed computing or would like to refresh your understanding of it.

Chapter 1 provides overview information; it is not intended to be a comprehensive discussion of distributed computing. You should have already read *z/OS DCE Introduction* for a more complete understanding.

Chapter 2, "z/OS DCE," is an overview of the z/OS DCE and Application Support architecture and technology components. This is the IBM implementation of OSF DCE on the z/OS platform. Read this section to become familiar with the technology behind z/OS DCE.

Chapter 3, "Building DCE Disciplines," discusses how proven, existing, management disciplines can be valuable tools for a workstation user in a distributed computing environment.

Chapter 4, "Planning Your Distributed System," assists you in the planning for the configuration and maintenance of z/OS DCE. A scenario helps answer the questions involved in planning for the implementation of a distributed system.

Chapter 5, "Additional Planning Considerations," discusses some more complex planning topics.

Chapter 6, "Planning For Installation," outlines the steps in planning for the installation of z/OS DCE.

Conventions Used in This Book

This book uses the following typographic conventions:

Bold	**Bold** words or characters represent system elements that you must enter into the system literally, such as commands, options, or path names.
Italic	*Italic* words or characters represent values for variables.
`Example font`	Examples and information displayed by the system appear in `constant width type style`.

[ ] Brackets enclose optional items in format and syntax descriptions.

{ } Braces enclose a list from which you must choose an item in format and syntax descriptions.

| A vertical bar separates items in a list of choices.

< > Angle brackets enclose the name of a key on the keyboard.

... Horizontal ellipsis points indicate that you can repeat the preceding item one or more times.

\ A backslash is used as a continuation character when entering commands from the shell that exceed one line (255 characters). If the command exceeds one line, use the backslash character **** as the last non-blank character on the line to be continued, and continue the command on the next line.

This book uses the following keying conventions:

<**Alt-***c*> The notation <**Alt-***c*> followed by the name of a key indicates a control character sequence.

<**Return**> The notation <**Return**> refers to the key on your keyboard that is labeled with the word Return or Enter, or with a left arrow.

Entering commands When instructed to enter a command, type the command name and then press <**Return**>.

Where to Find More Information

Where necessary, this book references information in other books using shortened versions of the book title. For complete titles and order numbers of the books for all products that are part of z/OS, see the *z/OS Information Roadmap*, SA22-7500. For complete titles and order numbers of the books for z/OS DCE, refer to the publications listed in the "Bibliography" on page 67.

For information about installing z/OS DCE components, see the *z/OS Program Directory*.

For a description of DCE's technology components: from a high-level overview to a discussion of individual components and their interdependencies, read the *z/OS DCE Introduction*, GC24-5911.

Softcopy Publications

The z/OS DCE library is available on a CD-ROM, *z/OS Collection*, SK3T-4269. The CD-ROM online library collection is a set of unlicensed books for z/OS and related products that includes the IBM Library Reader.™ This is a program that enables you to view the BookManager® files. This CD-ROM also contains the Portable Document Format (PDF) files. You can view or print these files with the Adobe Acrobat reader.

Internet Sources

The Softcopy z/OS publications are also available for web-browsing and for viewing or printing PDFs using the following URL:

```
http://www.ibm.com/servers/eserver/zseries/zos/bkserv/
```

You can also provide comments about this book and any other z/OS documentation by visiting that URL. Your feedback is important in helping to provide the most accurate and high-quality information.

Using LookAt to Look up Message Explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages. You can also use LookAt to look up explanations of system abends.

Using LookAt to find information is faster than a conventional search because LookAt goes directly to the explanation.

LookAt can be accessed from the Internet or from a TSO command line.

You can use LookAt on the Internet at:

```
http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html
```

To use LookAt as a TSO command, LookAt must be installed on your host system. You can obtain the LookAt code for TSO from the LookAt Web site by clicking on the **News and Help** link or from the *z/OS Collection*, SK3T-4269.

To find a message explanation from a TSO command line, simply enter: **lookat** *message-id* as in the following:

```
lookat iec192i
```

This results in direct access to the message explanation for message IEC192I.

To find a message explanation from the LookAt Web site, simply enter the message ID and select the release with which you are working.

Note: Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*, SA22-7624. For such messages, LookAt prompts you to choose which book to open.

Accessing Licensed Books on the Web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link site:

```
http://www.ibm.com/servers/resourcelink
```

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link user ID, password, and z/OS licensed book key code. The z/OS order that you received provides a memo that includes your key code.

To obtain your IBM Resource Link user ID and password, logon to:

```
http://www.ibm.com/servers/resourcelink
```

To register for access to the z/OS licensed books:

1. Logon to Resource Link using your Resource Link user ID and password.
2. Select **User Profiles** located on the left-hand navigation bar.
3. Select **Access Profile**.
4. Select **Request Access to Licensed books**.
5. Supply your key code where requested and select the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed.

After your request is processed you will receive an e-mail confirmation.

Note: You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books:

1. Logon to Resource Link using your Resource Link user ID and password.
2. Select **Library**.
3. Select **zSeries**.
4. Select **Software**.
5. Select **z/OS**.
6. Access the licensed book by selecting the appropriate element.

Chapter 1. Introducing Distributed Computing

Distributed computing is built upon the principle of functional distribution. This means that not all processing functions are performed by a single system; rather, they are distributed to multiple systems, to exploit each system's specialized services or computing strengths.

Distributed computing involves the cooperation of two or more systems communicating over a network. Different hardware architectures and operating system software are connected by various network protocols. The variety of different hardware, software, and applications make this network connection complex. Figure 1 shows an example of the complexity of such a distributed environment. The larger the number of different operating systems, application environments, and network protocols, the more difficult it is to tie these entities together and to manage them.

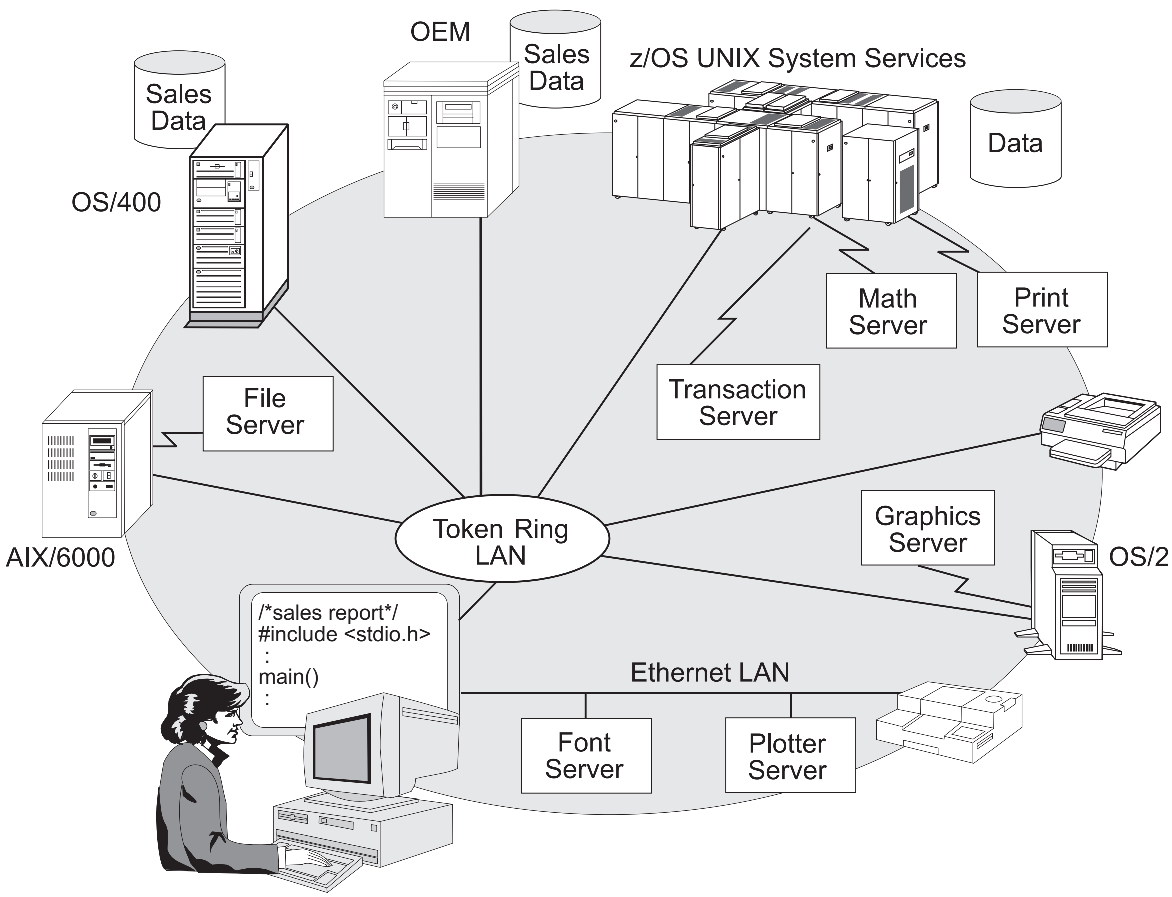

Figure 1. Today's Complex Computing Environment

The complexities of today's environment also make it very difficult to link resources from an application standpoint. Programmers can write applications that cooperate across machines by explicitly writing the code that performs the network communications, but this task requires much time, expertise, and expense. Wherever the application programmer must write code to overcome the differences between systems or network protocols, the risk of error increases.

The time, skill, and expertise required to develop and manage programs on one system are not easily transferred to other systems. Programmers and administrators, who are experts on one system, may not be as efficient working with or connecting to another system. Distributed computing in today's complex environment may be costly to implement. These difficulties can be overcome by moving to a Distributed Computing Environment (DCE).

DCE is a set of services that support the creation, use, and maintenance of distributed applications that use processes on different hardware platforms and on different operating systems. With the DCE architecture, programmers can develop and run distributed applications in this heterogeneous computing environment. Figure 2 illustrates how DCE masks physical differences between systems and networks from the application programmer, who can then concentrate on writing applications that use existing resources regardless of their location. DCE's services provide the support required in a distributed system like the support an operating system provides in a centralized system.

DCE supports both very large networks with many users, as well as smaller networks.

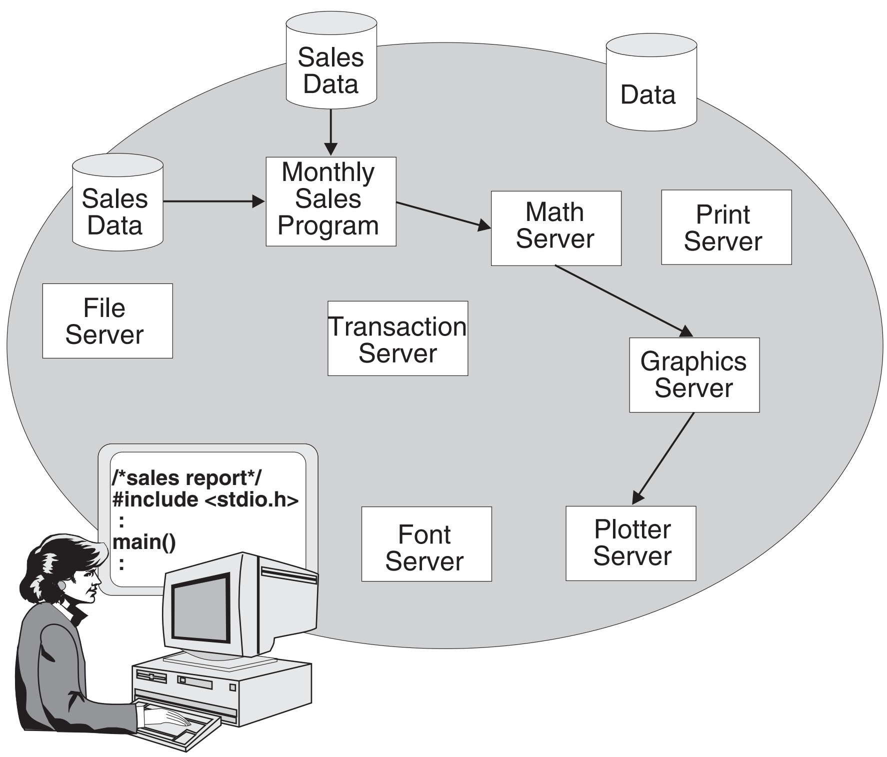

Figure 2. What the Application Programmer Sees with DCE

The end user, who for example wants to monitor sales, is unaware of the underlying complexities of her computing environment. Figure 3 shows the end user printing a sales report. All she knows or needs to know is how to access and manipulate the sales information and print out reports.

The end users should have maximum flexibility with minimal explicit knowledge of the environment. Applications do not expose them to the underlying physical complexities.

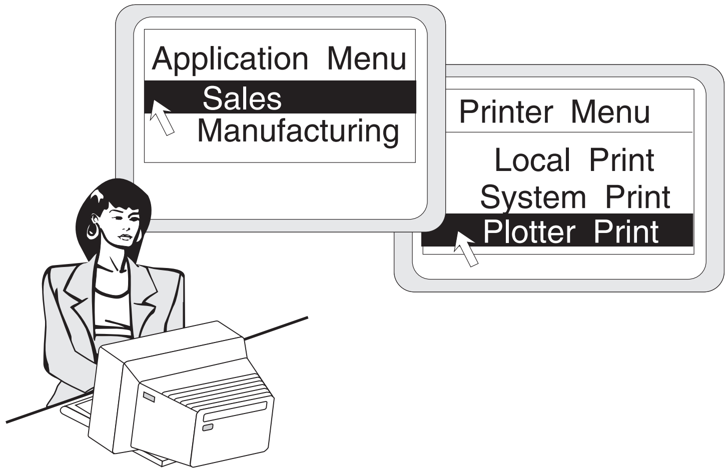

Figure 3. End User's View

Remember that, behind the simplicity of the end user's view, lies the complex computing environment shown in Figure 1.

The DCE services also provide a set of tools that support management tasks that system administrators need to perform. They can manage DCE without having to know about system internals.

Clients and Servers

DCE is based on the client/server model. The client side is a machine or process that initiates a request and receives the benefit of a specialized service. The server side is a machine or process that provides the requested specialized service.

A distributed application consists of a client side that initiates a request for a service, and a server side that receives and processes the request, and returns any results to the client. For example, a client may request that a file be printed, and the server where the printer resides carries out that request.

A machine can be both a client and a server. In processing the print request, the print server may ask a file server to supply a copy of the file for printing, as shown in Figure 4.

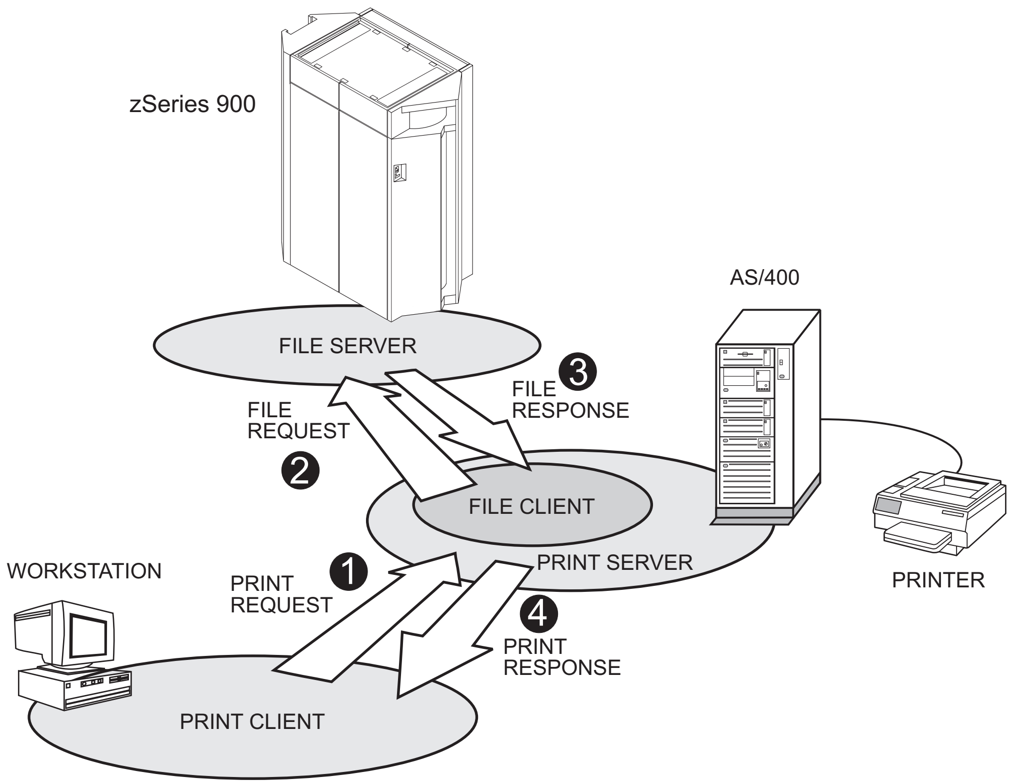

Figure 4. Client/Server Roles

In general, a server node is specialized. It performs a function that is only found on that particular machine. Client nodes are generalized. Typically, many nodes can be clients of a specialized service, and they can also be clients for many different services.

Cells

A cell is DCE's basic unit of organization. It is an administrative domain that allows distributed management of users, machines, and resources that share a common purpose. Members of an organization who are working on the same project are likely to belong to the same cell. In a large organization with several cells, the sales team could belong to one cell and the finance team could belong to a second. A small organization might only have one cell for both sales and finance, because they share the same financial information and the same level of security. Members of a cell are usually located in a common geographic area, but they can be located in different buildings, different cities, or even different countries, providing good connectivity exists.

The Open Software Foundation

The Distributed Computing Environment can trace its beginnings to the Open Software Foundation (OSF). Its goal is to encourage the development of an open, multivendor computing environment. The vehicle for delivery of this computing environment is DCE. The underlying vision of OSF is application-level interoperation among many heterogeneous systems to form a distributed system.

Various major computer vendors (IBM included) have contributed their expertise and proven technologies for inclusion in OSF DCE. A major benefit of OSF DCE is that it is composed of established running code. Not merely the specification of an evolving standard, it is a present-day solution.

OSF DCE provides a high-level, coherent environment for developing and running applications on a distributed system. It provides tools for developing distributed applications and services for running applications.

OSF DCE is oriented toward heterogeneous systems. Although a distributed system can be implemented using a single operating system running on all nodes in the network, the architecture allows for different operating systems and different hardware platforms. Applications built using DCE can be ported to other platforms running DCE.

DCE supports the sharing of data throughout the distributed system. Authorized users can access data regardless of their operating system, hardware platform, or location in the distributed system.

OSF Distributed Computing Environment

The OSF Distributed Computing Environment (OSF DCE) architecture defines a set of services layered between the operating system and network and the distributed applications. It provides the services that enable a distributed application to interact with a collection of either heterogeneous or homogeneous computers, operating systems, and networks as if it was a single system. Figure 5 on page 6 shows the OSF DCE architecture and its technologies, along with their relationship to applications, underlying system support, and placeholders for future services.

The technologies fall into two general categories: programming services and distributed services.

The DCE Threads Service and DCE Remote Procedure Call (RPC) are programming services; they include libraries that implement application program interfaces (APIs) and program development tools.

The remaining technologies are distributed services. Each consists of a daemon, or server process, that runs continuously on a machine and responds to requests sent over the network. These services have administrative components to manage the service, and have APIs that programmers can use to access these components.

Application programmers deal mostly with the programming services. Although the distributed services are accessed through APIs, the programmer usually only uses them indirectly through RPC, which in turn uses the APIs of the distributed services. Administrators, on the other hand, deal mostly with the distributed services, because these have significant management requirements.

The administration of the OSF DCE components can itself be distributed, allowing distributed management of the distributed computing environment. The need for a centralized authority is alleviated making the distributed environment more flexible, scalable, and allowing for redundancy. You can maintain a central authority, distribute authority, or implement a combination of both.

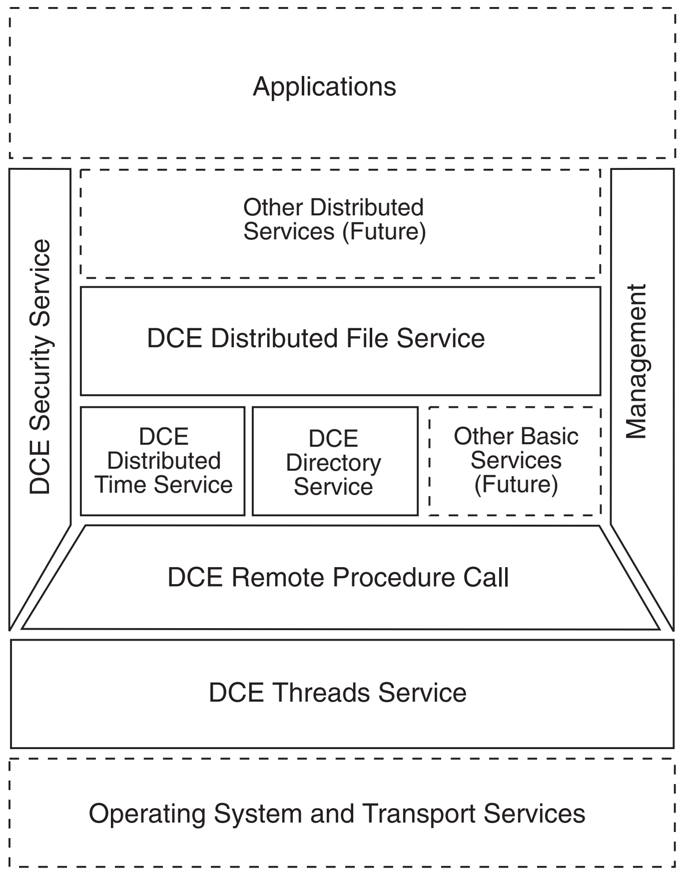

Figure 5. OSF DCE Services

The following sections briefly describe DCE services. For more detailed information about the DCE architecture and its components, refer to the *z/OS DCE Introduction*.

Remote Procedure Call: The DCE Remote Procedure Call (RPC) is a service for calling a procedure on a remote machine as if it were a local procedure call. Based on the client/server model, it allows application programmers to extend the local procedure call to a distributed environment. The application programmer does not have to be concerned with the details of network communications between client and server nodes.

Programmers using RPC do not need to rewrite applications to port them to different architectures, operating systems, or communication protocols. RPC hides communication details and removes system and hardware dependencies.

An end user does not see any of the client/server interaction and does not know or need to know if procedures are local or remote.

Threads Service: Many computer programs are designed to run sequentially, with only one point in the program currently in execution. However, some computer programs lend themselves to being structured as multiple flows of control and perform better when they contain multiple threads of control.

In distributed computing, RPC enables the use of multiple threads of control. When a client issues an RPC, it blocks (or waits) until a response is returned from the server. If a client uses multiple threads of control, work can continue in another thread while the thread awaiting an RPC response is blocked. Because servers can also issue RPCs, a similar scenario applies. A separate thread can also handle each client request. While one server thread is waiting for an input or output operation to finish, another server thread can continue working.

Directory Service: A distributed system may contain many users, machines, and other resources, along with large amounts of data, all geographically dispersed. The goal of a directory is to provide up-to-date addressing information for network resources. Users can identify, by name, resources such as servers, files, disks, or print queues, and gain access to them without needing to know where they are located. The sharing of information is based on unique names, not on location.

The DCE Directory Service is a distributed and replicated service. It is distributed because the information that forms the database is stored in different places. Information on one group of users and resources can be stored on one directory server, while information about a second group of users and resources is stored on a different directory server. The Directory Service can replicate information, storing it in more than one location, making it readily available. Performance is enhanced because directory information can be replicated close to its users.

The Directory Service consists of two types of directory services: one that manages resources within a cell, and a global name service that manages resources outside the local cell.

The Cell Directory Service (CDS) stores and manages the names and attributes of resources in a cell. There are two global name services that can be used to locate resources outside the local cell: Global Directory Service (GDS) and Domain Name System (DNS). GDS is based on the CCITT X.500/ISO 9594 international standard and so is positioned to participate in the anticipated worldwide X.500 directory service. DNS, used by the Internet to track the vast number of machines that comprise the Internet, provides an alternative means for independent cells to locate and interact with each other.

Distributed Time Service: The DCE Distributed Time Service (DTS) provides synchronized time on the computers participating in a distributed computing environment. In a single system, one clock provides the time to all applications. In a distributed system, however, each node has its own clock. Even if all the clocks in a distributed system could be set to one consistent time at some point, they would drift away from that time at different rates. As a result, different nodes would have different times. This is a problem for distributed applications where the ordering of events is important. DTS enables distributed applications to determine event sequencing, duration, and scheduling. DTS synchronizes a DCE host's time with Universal Coordinated Time (UTC), an international time standard.

Security Service: The DCE Security Service provides trustworthy identification and certification of principals (users, clients, servers, and systems), offers integrity and privacy of communications, and enables controlled access to resources. It controls the interaction between clients and servers.

Today, most systems provide one way authentication, where the client proves its identity to the server. Server identity is rarely verified. In a distributed environment, this trust of servers may be lessened, leading to a requirement for two-way authentication.

In two-way authentication, each server must be able to verify the identity of each client, and each client must be confident that it is communicating with a secure server. Clients and servers use trusted keys to request and provide services. Each server must maintain trusted key information for each client that it can serve, and every client must know a trusted key for each server it might use. Two-way authentication is difficult to administer. Every time a server's information changes, all its clients must be updated.

DCE simplifies administration and adds security by implementing a trusted-third-party approach based on the Kerberos technology. The security server, acting as a trusted third party, maintains the trusted key information. Clients and servers no longer need to store this information. The Security Server identifies and certifies principals (authentication) and provides information on the privileges associated with each principal. Privileges enable servers to perform selected operations (authorization) for authenticated principals.

Note: In this book the term "DCE Security Server" (or simply "Security Server") refers to the z/OS SecureWay® Security Server DCE or to a DCE Security Server provided on another host in the DCE cell. The z/OS SecureWay Security Server DCE is a component of the SecureWay Security Server for z/OS.

RACF® Interoperability and Single Sign-on: z/OS DCE also provides interoperability between Resource Access Control Facility (RACF), a component of the SecureWay Security Server for z/OS, and DCE. This security interoperability allows a DCE client to access a DCE-enabled server on a z/OS system and allows the DCE server to acquire corresponding local security credentials for the DCE client to access z/OS resources. The interoperability function allows:

- Appropriately authorized DCE servers to acquire corresponding z/OS security credentials for the DCE client and to use the DCE client's corresponding z/OS user ID for access to RACF-authorized resources.
- A z/OS user to be transparently logged in to DCE when necessary, without prompting for a DCE user ID or password. This ability is called single sign-on. With this feature, a z/OS user authenticates to z/OS and can run a DCE program without reauthenticating to DCE.

Integrating the DCE Components

Figure 6 on page 9 shows how the DCE components work together in a distributed environment.

1. Distributed Time Service (DTS) keeps the clocks on the different machines synchronized.
2. Each server is registered with the Directory Service. The server functions (Procedure X and Procedure Y) advertise their locations (host address and network identification) in the Directory. The server also authenticates itself with the Security Service.
3. The client's RPC queries the Directory for a server offering Procedure X, which is required or called in Application 1.
4. The Security Service verifies the identity and authority of the client.
5. The client (Application 1) makes an RPC call to the server for Procedure X.
6. The server checks the client's security credentials.
7. The server performs Procedure X.
8. The server returns the results to the client application.

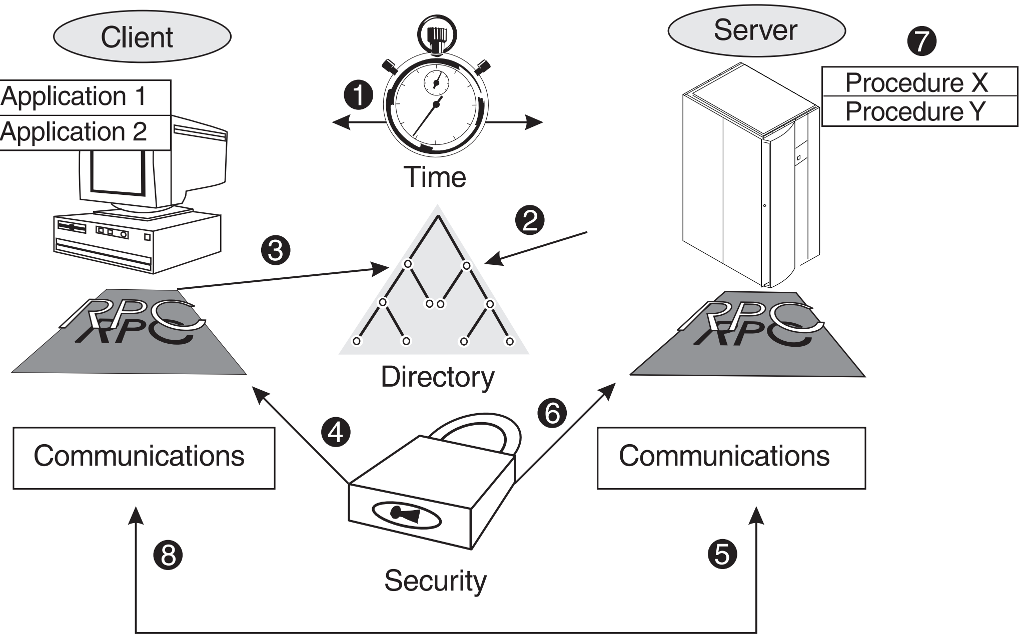

Figure 6. DCE's Integrated Components

The DCE components, although running in different locations, are so well integrated that users do not know or need to know if the applications are running locally (on their machine) or remotely.

This integration also hides the details of security and directory from the application programmers using them. Applications written on one machine can run on other machines. The location of required or called procedures is handled by the directory. The portability of the code, the hidden details of security and the directory, and the transparency of the communications interface ensures easy platform integration and flexibility in design for programmers who are creating distributed application programs.

Chapter 2. z/OS DCE

The implementation of the OSF DCE technology as part of z/OS UNIX System Services is a continuation of the IBM strategy of open systems and interoperability across different platforms. This open and distributed strategy views z/OS both in the roles of server to other client systems and as a peer with other server systems in the network. This provides customers with IBM products that work with products from other software vendors.

Architecture

The z/OS DCE runtime environment is made up of the base DCE functions or services described in "OSF Distributed Computing Environment" on page 5. The runtime environment does not require all the services to be installed on a single machine; rather, they can be located on various machines throughout your DCE environment. Not all vendor implementations of DCE fully support all the base functions.

z/OS DCE support for the base functions falls into three categories: full support, client-only support, and no support. The following sections detail z/OS DCE support. If your environment includes the base functions, you can successfully write a full DCE application on z/OS, as it will have access to the services available on other machines.

Figure 7 on page 12 summarizes the services available with z/OS DCE. These include:

- Remote Procedure Call
- Directory Service
- Distributed Time Service
- Threads Service
- Security Service.

IBM z/OS DCE Application Support is an optional z/OS DCE product that provides access to existing data and application logic in CICS® and IMS™ through the following:

- z/OS DCE Application Support CICS
- z/OS DCE Application Support IMS

This selection of DCE services enables a closer coupling of existing IBM subsystem services, as well as enabling IBM systems to function as parts of a heterogeneous distributed system.

The following sections describe the DCE services available for z/OS DCE. Figure 9 on page 17 summarizes the DCE processes that can be installed on a z/OS host.

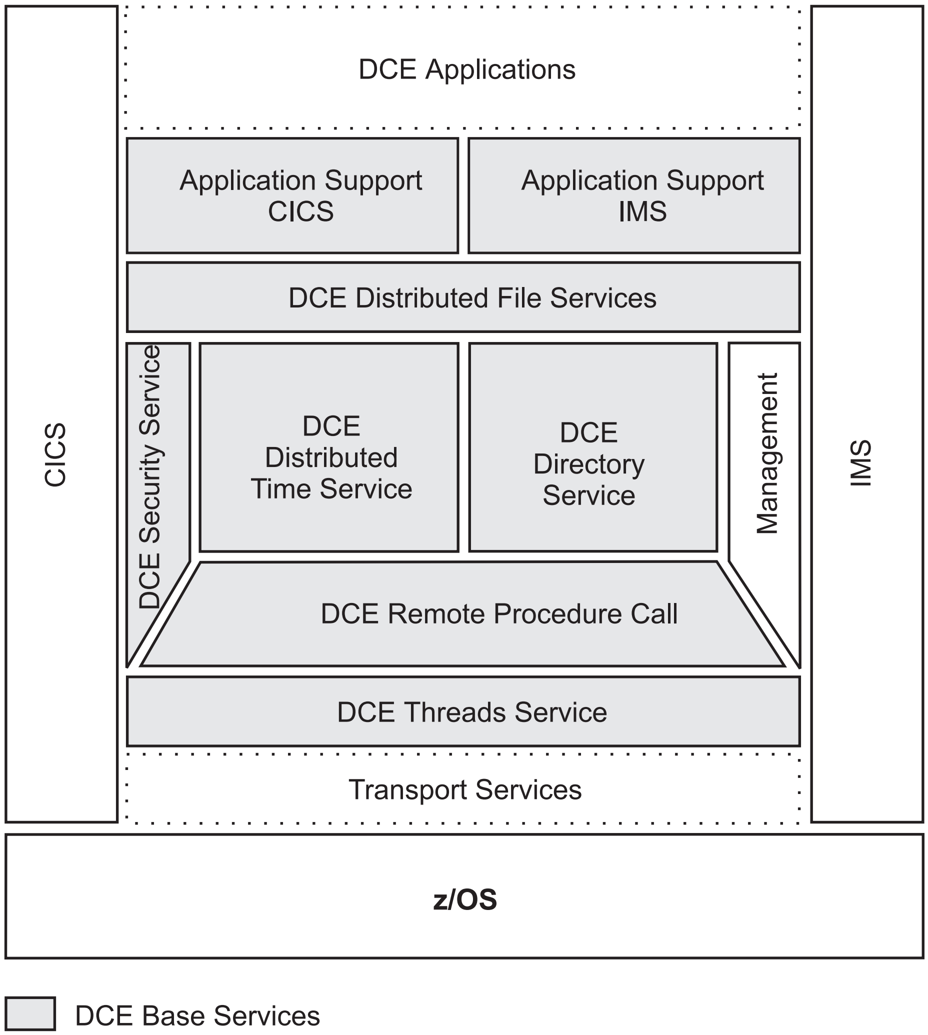

Figure 7. z/OS DCE Architecture

Remote Procedure Call

RPC is the primary method for client-to-server communication. The DCE Host Daemon (DCED) is the server process that supports the use of RPC services on a host.

For a client program to send an RPC request, it must be able to locate a server that can service the request and the address of the process on that server. First, the client uses the Directory Service to find an appropriate server. It then uses the DCE Host Daemon to locate the required process or interface on that server machine.

DCE Host Daemon provides the Endpoint Map Service, which enables a client to locate the network endpoint or address of the process that it requires. This Endpoint Map Service gives DCE flexibility. Clients do not need to know the location or endpoint of an interface before using it. DCE Host Daemon's Endpoint Map Service enables RPC-based clients and servers to communicate without needing to know interface endpoints. Every DCE server machine must run DCE Host Daemon.

RPC also handles any data conversions, such ASCII to EBCDIC, that might be required between unlike machines in a heterogeneous computing environment.

z/OS DCE fully supports DCE RPC, making z/OS DCE applications and data available throughout your distributed computing environment.

Using the SecureWay Communications Server

See the *z/OS DCE Administration Guide* for some DCE-related considerations regarding use of SecureWay Communications Server for z/OS.

Directory Service

The DCE Directory Service provides directory service at the cell and the global levels.

Cell Directory Service

CDS stores and manages the names and attributes of resources in a cell. It consists of a CDS server and a CDS clerk (client).

Each resource name and its associated attributes are stored as an entry in a database of directory information. The CDS server maintains this database and handles requests to create, modify, and look up resource information.

The CDS clerk is installed on a client machine. It serves as the intermediary between client applications and CDS servers. Its most important function is maintaining a cache of information obtained through directory queries. The clerk stores responses to queries in its cache so that the next time a similar query is made, the information is already available to the client, and no network communication with the CDS server is necessary.

The CDS advertiser is responsible for sending and receiving information or advertisements of the presence and status of CDS servers on DCE hosts. It is also responsible for creating the cache used by the CDS clerk. And periodically saving the cache to disk.

z/OS DCE implements the CDS clerk, advertiser, and server. The DCE Control Program (DCECP) available with z/OS DCE is the interface you use to control CDS servers from z/OS, whether they are on z/OS or non-z/OS hosts.

Global Directory Service

The Global Directory Service (GDS) is used to look up a name outside of the local cell. It is the connection that enables independent cells to find out about and interact with one another. GDS is based on the CCITT X.500/ISO 9594 international standard and therefore can participate in the X.500 directory service.

When a CDS server receives a directory request, it determines whether it can find the object in its own cell or whether it needs to contact another server to help locate the object. If the object is located in another cell, the CDS server in that cell must be used to resolve the name. To contact the CDS server in another cell, the local CDS server must know how to reach the other cell. The Global Directory Agent (GDA) assists CDS in determining the information it needs to contact another cell. The GDA acts as an intermediary between CDS and GDS.

Although z/OS does not support GDS, it does provide a GDA daemon. The GDA is used in DCE implementations where communications with directory services in other cells are required.

z/OS DCE can also use the name-to-address translation services of the Domain Name System (DNS), that is used by the Internet to keep track of the large set of machines that comprise the Internet. DNS provides an alternative means for independent cells to locate and interact with each other.

Distributed Time Service

The Distributed Time Service (DTS) provides synchronized time to the computers participating in DCE. It checks time synchronization and adjusts clocks when the clock error exceeds the range that you have set. This clock synchronization enables distributed applications to determine event sequencing, duration, and scheduling.

The time clerk is the client side of DTS. It keeps a machine's local time synchronized by asking time servers for the correct time and adjusting the system clock accordingly.

A time server is a node that answers time queries. Time clerks query the time servers, but time servers also query each other, computing system time and adjusting their own clocks when appropriate.

There are three types of time servers:

- Local Time Servers

  Local time servers are on the same LAN as the requesting clerk. Local time servers maintain their clocks by synchronizing with each other.

- Global Time Servers

  Local time servers are only available to servers and clerks on a single LAN, but global time servers are available across multiple LANs, extended LANs, or WANs. It may be desirable to have local time servers synchronizing with a global server rather than having time synchronization occurring exclusively in a LAN.

- Courier Time Servers

  A courier time server is needed when synchronizing with a global time server. Couriers use the responses from both local and global time servers when synchronizing their own clocks.

The DTS time provider interface synchronizes system clocks with the UTC time standard. This time is available through radios, satellites, and telephones. Connection to these external time providers is optional.

z/OS DCE implements DTS, but does not support connections between z/OS and external time providers. However z/OS DCE provides a null time provider program (NTP) that can be used by z/OS hosts that have been configured as DTS servers. A null time provider assumes that the host has a reliable time source and so does not query external time sources. If NTP is configured, the z/OS DTS server will not take the time from other DTS servers in the cell, but DTS clerks (non-z/OS) can still use the z/OS-time when they synchronize.

Security Service

The Security Service enables clients and servers to prove their identities to each other. It provides integrity and privacy of communications and supports controlled access to resources. The DCE Security server consists of three cooperative services:

- Registry Service

  The Registry Service manages a cell's security database. This database contains entries for DCE principals, which can be users, servers, or machines.

- Authentication Service

  This service provides trustworthy identification of the principals involved in communications.

- Privilege Service

  The Privilege Service forwards, in a secure way, the information that a server needs to know to determine if a principal has the right or authority to do what it wants to do.

The Security Service also includes two facilities:

- Access Control List Facility

  DCE Access Control Lists (ACLs) are lists of users who are authorized to access given resources.

- Login Facility

  The Login Facility initializes a user's DCE security environment. It authenticates the user to the Security Service by means of the user's password.

  If your installation will have many DCE clients logging into DCE from z/OS, the performance of these logins can be improved by using a Temporary File System (TFS). See the *z/OS DCE Administration Guide* for how to use a TFS.

Security and RPC cooperate to provide authentication, authorization and secure communications. The user security environment set up by the Login Facility combines with the server's registry of its name and the authentication that it supports to provide secure RPC.

RACF Interoperability and Single Sign-on

z/OS DCE also provides interoperability between Resource Access Control Facility (RACF) on z/OS and DCE. This security interoperability allows a DCE client to access a DCE-enabled server on a z/OS system and allows the DCE-enabled server to acquire corresponding local security credentials for a DCE client to access z/OS resources. The interoperability function allows:

- Appropriately authorized DCE servers to acquire corresponding z/OS security credentials for the DCE client and to use the DCE client's corresponding z/OS user ID for access to RACF-authorized resources.
- A z/OS user to be transparently logged in to DCE when necessary, without prompting for a DCE user ID or password. This ability is called single sign-on. With this feature, a z/OS user authenticates to z/OS and can run a DCE program without reauthenticating to DCE.

z/OS DCE also provides interoperability utilities. These incorporate into RACF the information that associates a z/OS user ID with a DCE principal's identifying information and the DCE principal's UUID with the corresponding z/OS user ID. This is called cross-linking information and is what allows interoperability and single sign-on to work.

The cross-linking information must be set up before interoperability functions can be used. To do this, DCE provides two utilities, **mvsimpt** and **mvsexpt**, for creating the initial cross-linking between the two registries. This cross-linking can be done from either the RACF database or the DCE registry, but **mvsimpt** and **mvsexpt** must be run from the z/OS system where the RACF database resides whose users are to be cross linked.

z/OS also provides application programming interfaces so that you can write your own server programs to take advantage of RACF interoperability.

Accessing CICS and IMS

DCE access to Customer Information Control System (CICS) and Information Management System (IMS) programs and data is available with the IBM z/OS DCE Application Support product. With the Application Support servers, an enterprise can extend the strengths of the CICS and IMS processing and development beyond their traditional z/OS environment and into the world of open systems.

DCE AS CICS and DCE AS IMS provide workstation integration. The DCE client can be non-CICS, non-IMS, or non-IBM. The client can access multiple CICS and IMS regions and other z/OS and non-z/OS applications and data at the same time.

Figure 8 illustrates transparent access to the RPC client through the z/OS DCE Application Support CICS Feature, and z/OS DCE Application Support IMS Feature.

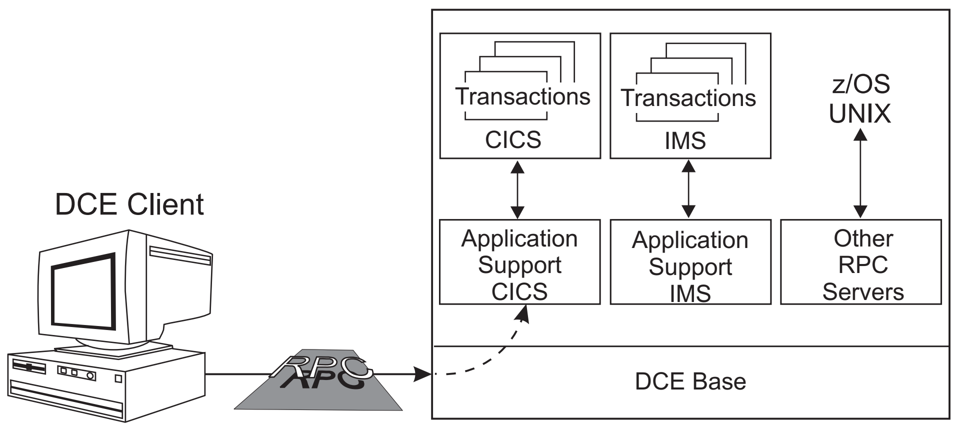

Figure 8. Application Support server

Using the client/server paradigm, think of CICS and IMS transaction programs as application servers, that provide the services requested by a DCE client.

Application Support:

- Provides access to existing data and transaction logic in CICS and IMS
- Uses the DCE Directory Services to provide resource location transparency and reduced administration
- Uses the DCE Security Service to provide secure access from any DCE client in the network
- Builds on existing skills:
  - Mainframe developers can continue to write COBOL transactions in the familiar CICS and IMS environments
  - Workstation developers can continue to write client programs using RPC calls
- Supports COBOL data types to ease description of COBOL transaction interfaces

z/OS DCE Implementation

One of the advantages of DCE is that its services can work with each other. Whenever appropriate, DCE services make use of other DCE technologies to accomplish their tasks. For example, DTS uses RPC to handle the communications between its time clerks and time servers. The z/OS DCE implementation of the DCE technologies is somewhat similar. A subset of the DCE processes and protocols can be installed on a z/OS host, while others must be located on other hosts in your cell.

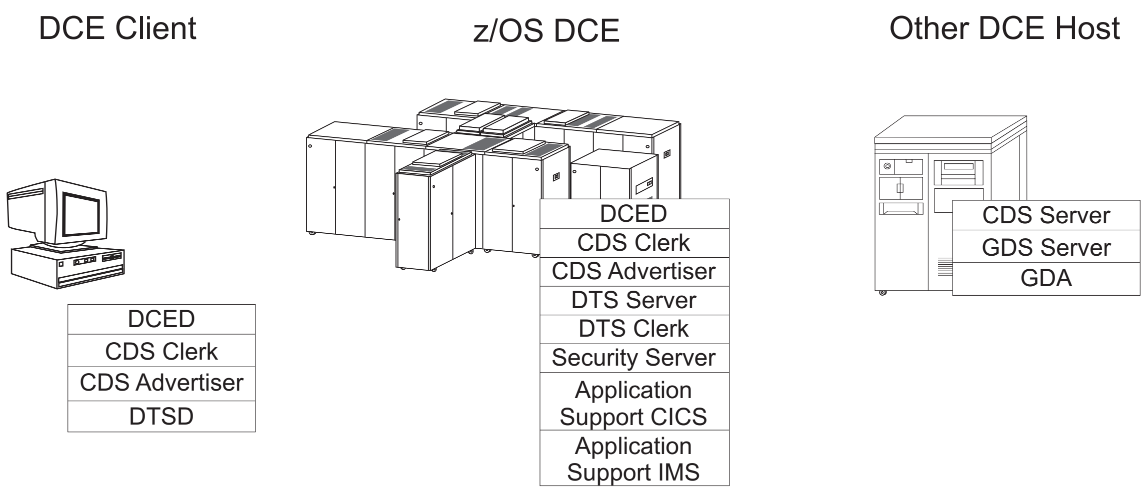

Figure 9. z/OS Support of DCE Services

Distributed computing means functional distribution. A DCE runtime environment requires that a cell must have access to all the DCE services, but the distributed nature of DCE means those functions can be distributed throughout your cell. They do not need to reside on a single host. Regardless of the location of DCE services z/OS DCE enables z/OS as a full participant in single-cell and multicell configurations of DCE.

Chapter 3. Building DCE Disciplines

Traditionally, computers have been independent centralized systems, consisting of a central processor connected to peripherals such as storage units, display monitors, and other input and output devices. Often much of the equipment was locked behind the glass walls of a data center.

Although appealing in terms of cost and control, these systems often lacked responsiveness. The people who used them (financial, engineering, sales) often had conflicting needs. The installation of inexpensive, stand-alone, departmental minicomputers and workstations was an attempt to return responsive computing to the user community. But as the number of computer systems in the office grew, their users soon realized they needed to share data and resources among them.

While the biggest obstacle to computer cooperation has been the lack of compatibility between systems, there can also be incompatibility between users. The guidelines for use of a computer system can range from strictly enforced controls to no rules at all. A DCE environment allows individual systems to share data, processors, applications, and devices and operate as a single unit, but a portion of the quality of that environment is dependent on the disciplines practiced by its users.

Glasshouse disciplines are those practices developed in data centers over the past several decades. They provide value to the customer by enhancing the reliability, availability, and serviceability of the computing environment. Today's data centers operate for months without interruption. When interruptions do occur, they are pre-scheduled for system or hardware maintenance. This stability is due in part to the quality of the hardware and software components, but if this were the total story then all data centers would be interruption-free. What separates one data center from another is its implementation of glasshouse disciplines.

Changing Environments

From a user's perspective, traditional computing environments are host-centered. That is, whenever users need to run applications, they log onto the host using a user ID and a password, and then call a procedure to run an application. Host-centered computing works well in single platform settings, but such environments are becoming quite rare. Users are more likely to need access to applications running on a combination of enterprise servers (formerly called mainframes), midsized business hosts, file servers, and desktop computers. Often the data from one application is needed in another application. One example might be the use of data from a CICS application running on an enterprise server. Because of the lack of interoperability between platforms, such data is often moved manually, a costly and dangerous practice.

DCE's interoperability allows applications to communicate with other applications, on other platforms, using RPC. Data and applications can be delivered to a user's desktop without the user having to know where the applications are running, or that multiple platforms might be involved. Applications appear as extensions to the user's desktop. This desktop-centered computing is secure, scalable, and open. It allows the strengths of each operating system to be brought together to provide solutions that make business more efficient and competitive.

An example of a distributed application might involve three basic service platforms:

- An enterprise server system used for data storage
- A supercomputer used as the computation processor
- Desktop computers capable of three dimensional graphics.

In a traditional host-centered environment, the user would:

- Login to each platform
- Move data from the enterprise server to the supercomputer
- Check to make sure that the data was moved successfully
- Run an application on the supercomputer to process the data
- Check for successful completion of the process before moving the results to the workstation for viewing.

After this application is redesigned for DCE, the enterprise server contains a data access component that provides access to its data from other computers in the network. Now, the super computer can read the data directly and periodically sends the results to the workstation for viewing.

The rest of this chapter uses this example to describe how glasshouse disciplines can be used to ensure the quality of this application.

Glasshouse Disciplines

Whether they are in a traditional data center supporting hosts and terminals or in a distributed computing environment supporting heterogeneous platforms, users expect the level of quality the following glasshouse disciplines provide:

Security

Security is of utmost concern in data centers. Methods of ensuring a secure environment include:

- Unique identifiers and expiring passwords for each user
- Selected tasks that can only be started by system operators
- Data secure through access control lists
- Periodic audits of all components.

Basically every resource has access locks.

You need to provide this same level of integrity for distributed applications. In the sample application, the user at the workstation obtains security credentials by logging onto the network with a user ID and a password (authentication). These credentials are embedded in RPC calls to applications on other platforms. Secure RPCs provide access to the data on the mid-range and vector processing on the enterprise server. Servers, like the vector application and the data access application, control access based on user credentials and their access privileges (authorization). The servers also have universal identifiers that they use to obtain credentials. Therefore both users and servers are authenticated in the environment.

Authentication and authorization are part of application design. Your organization must mandate how they should be applied. You need to develop a security model for application programmers and system administrators otherwise, as new applications are written, a hodgepodge of security practices will develop. A security model should include:

- Guidelines or requirements for securing client and server platforms and applications
- Good practices for programmers to follow when implementing DCE applications
- Information for system administrators
- Guidelines for designing audit plans.

Change Management

Change management as a discipline manages the introduction of change into the environment. In an enterprise server (mainframe) environment, where applications depend on 24 hour service, 7 days a week, introducing change to any component may cause a service outage. In an enterprise server environment, special procedures and practices have been developed to ensure that outages do not occur:

- Test environments are separate from production environments.
- Elaborate checklists are prepared identifying modules being changed.
- Changes are tested.
- A test is performed to ensure that the changes took place.
- Backout plans are developed in case the test fails.

A distributed environment requires a horizontal, cross platform view of change management. Changes on one platform can affect the services needed by another platform. Data centers are experienced in managing servers that they own, but managing change on desktop computers adds another level of complexity to change management. Now software distribution and versioning must be included in change management.

In the sample application, change management should be applied on each of the three platforms covering changes in hardware, system software, network, and DCE components. Any changes to the client-side or server-side of applications must also be managed.

Data Storage Management

Information is a business asset and needs to be protected against loss from user errors, hardware failures, and accidental or intentional corruptions. Backup and archiving are familiar facilities to administrators of enterprise server and midrange systems, but these might be new considerations for workstation users. The server platforms in the sample application should have their disk space managed to prevent system outages caused by out-of-disk space conditions.

Distributed Operations

Enterprise server environments typically employ operations staff who maintain the system and get it back up and running should any outages occur. Distributed operations present a new set of challenges. Operators and administrators will be responsible for network services. Computers may be housed nearby, but they could also be located in different buildings or even different cities. Personnel will have a larger variety of computers to manage and will initially need more help.

The servers in the example should be continually monitored. They include the hardware, operating system, and network. Additionally, the server applications should be monitored to keep them running. Provisions should be made for recovery in cases where the applications are found to be inactive. Performance thresholds for the services should be set and then monitored to provide insight for capacity planning and tuning.

Distributing Discipline

Distributed computing moves an organization away from **proprietary** standards toward a set of new open standards. With DCE, the strengths of the enterprise server can be exploited by users throughout the enterprise. This deployment should not overlook the rich set of disciplines currently practiced in the enterprise server world.

Chapter 4. Planning Your Distributed System

You will need to make a number of decisions when you are planning for the implementation of z/OS DCE. Some of these decisions are specific to z/OS DCE, but some apply to distributed computing in general.

This chapter presents you with a sample z/OS DCE planning scenario for the Vanroy Corporation, a fictitious US company based in Chicago. It details the various z/OS DCE and Application Support planning considerations and presents the decisions made by the Vanroy Corporation. You will have to make similar decisions when creating your distributed computing environment.

The Importance of Configuring z/OS DCE

For DCE to work properly, you must configure it. See the *z/OS DCE Configuring and Getting Started* book for instructions on how to configure DCE.

A Sample Cell

Detailed planning by the Vanroy Corporation resulted in the distributed computing environment illustrated in Figure 10 on page 24. The following sections describe the planning decisions made to create this environment. The Vanroy Corporation had to determine:

- The number of cells
- The size of cells
- The cell names
- The client requirements
- The server requirements
- The location of servers
- The replication of data
- The distribution of data
- The participation of nodes.

This chapter ends with a summary checklist of the z/OS DCE planning items considered in defining this distributed computing environment.

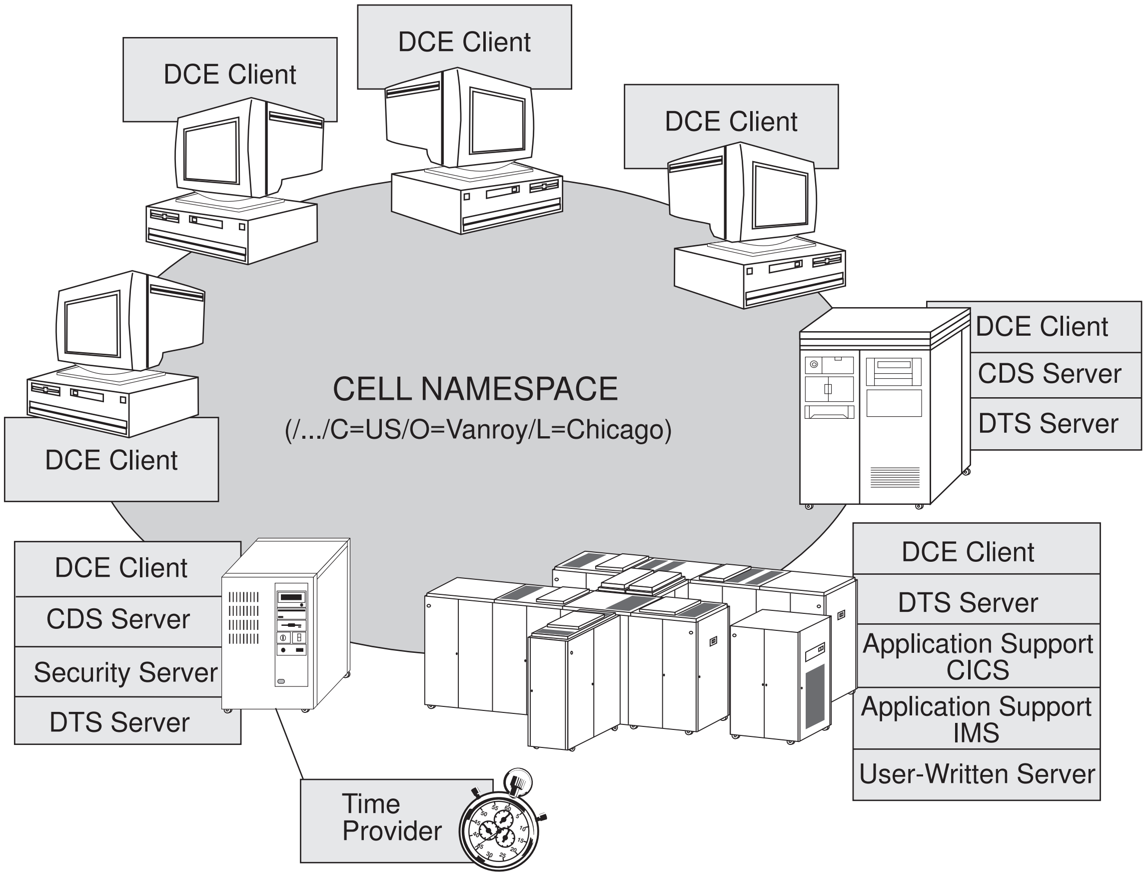

Figure 10. DCE Configuration for the Vanroy Corporation

Organizing the Cell Namespace

The boundaries of a cell are the limits of the cell namespace. The namespace is a hierarchical grouping of all the objects or names in a cell. Names identify the various resources in a cell, such as machines, databases, and users. An example of a cell namespace hierarchy can be found in “CDS Server” on page 29. A small organization can consist of a single cell, while a large company can have many cells. The main factors in determining cell boundaries are the common purpose and trust shared by a cell's principals. Principals within a cell belong to groups that share the same privileges. Members of a group share the same level of trust and are authorized to perform the same select set of actions. For example, the people developing a new product might form a single cell.

The cell namespace decisions you make, based on common purpose and shared trust, should be compared to your network set up. Comparing the physical boundaries of your cell to your LANs might suggest some minor changes to your cell namespace. Some advantages of matching a cell to a LAN are:

- Administration of both the cell and the LAN might be easier if the structures are similar.
- Configuration is easier if all DCE servers are on the same LAN.
- Performance is improved if all DCE servers are on the same LAN.
- CDS servers can broadcast their status to DCE clients on the same LAN.
- CDS can easily find replicas. If replicas are on other TCP/IP subnets, then manual administrative intervention is required.
- Manual administration is required for clients to be aware of servers on other subnets.

Because a set of administrative tasks is associated with setting up and maintaining each cell, it is reasonable to keep the number of cells to a minimum. However, the need of principals to share information is a more important consideration than administrative overhead.

The Vanroy Corporation is a small organization. Its computer hardware consists of four workstations, an enterprise server running z/OS, a UNIX-based OEM machine, and a smaller host running AIX/6000®. All hardware is on a single LAN. Its computer users share the same company information, with minimal data being restricted. Therefore, it is decided that a single cell will best meet the current distributed computing needs (as shown in Figure 11 on page 26).

Once you decide how many cells you need and where their boundaries will be, make an effort to keep your cell structure stable.

Establishing a Cell Name

Before you can configure your DCE cell, you need to establish a cell name. Choosing an appropriate cell name is important for the following reasons:

- Cells participating in the global namespace must have unique names that differentiate them from cells in other organizations.
- A uniquely identified cell name is critical to the operation of the security service. The name is the basis for authentication in your cell.
- Global cell names need to adhere to certain format standards.
- Changing a global name is complicated; do not select a temporary cell name.

If you plan to create a private cell or group of cells, and do not intend to interoperate with cells outside of your organization, you do not need to obtain a globally unique cell name. However, for your cell to communicate with cells outside of your organization, you need to obtain a globally unique cell name from the GDS or DNS naming authorities. The registration of your cell name must be completed before you can begin to configure the cell namespace. Remember, even if you do not initially use a global directory service to communicate with other cells, you may want to do so in the future.

Unique GDS Cell Names: If you want to interoperate with other cells in X.500 implementations in a global naming environment, you must contact the naming authority in charge of the entry under which you want to create a cell name.

For example, in the United States, the American National Standards Institute (ANSI) delegates names subordinate to the United States country (C=US) entry. If a US corporation is interested in participating in a worldwide X.500 directory, it would contact ANSI to reserve its corporate name as a unique organization name. If Vanroy is interested in participating in a worldwide X.500 directory, it would contact ANSI to reserve Vanroy as a unique US organization name (O=Vanroy).

Unique DNS Cell Names: If you plan to use DNS as your global name service, your cell name needs to follow the Internet Domain System conventions for site names. A cell name must contain at least two levels; for example, Vanroy's name could be vanroy.com. The name must end with the top-level domain suffix that indicates your type of institution:

.com	businesses and commercial organizations
.org	non commercial organizations
.edu	educational institutions
.gov	government institutions
.mil	military institutions

All top-level domains are then subdivided into second-level domains, which may be further subdivided into third-level domains, and so on, with each subdivision making the domain less general and more local. Although there is no restriction on the length of a name, a short name might prove more convenient to enter.

If you are already an Internet site, your Internet domain name is your DCE cell name. The naming authority for DNS names is the Network Information Center (NIC).

Initially, Vanroy will not be interoperating with cells outside of its organization, but because it may want to do so in the future, the decision is made to register the organization with ANSI. In its current single cell environment, the Vanroy cell name would be /.../C=US/O=Vanroy. But continuing with its view towards the future, Vanroy anticipates expanding to other US cities, resulting in multiple company cells. Unique cell names would be based on these US cities or locations (L=Chicago). The naming authority in the Vanroy company therefore establishes the initial cell entry as /.../C=US/O=Vanroy/L=Chicago. (See Figure 11.)

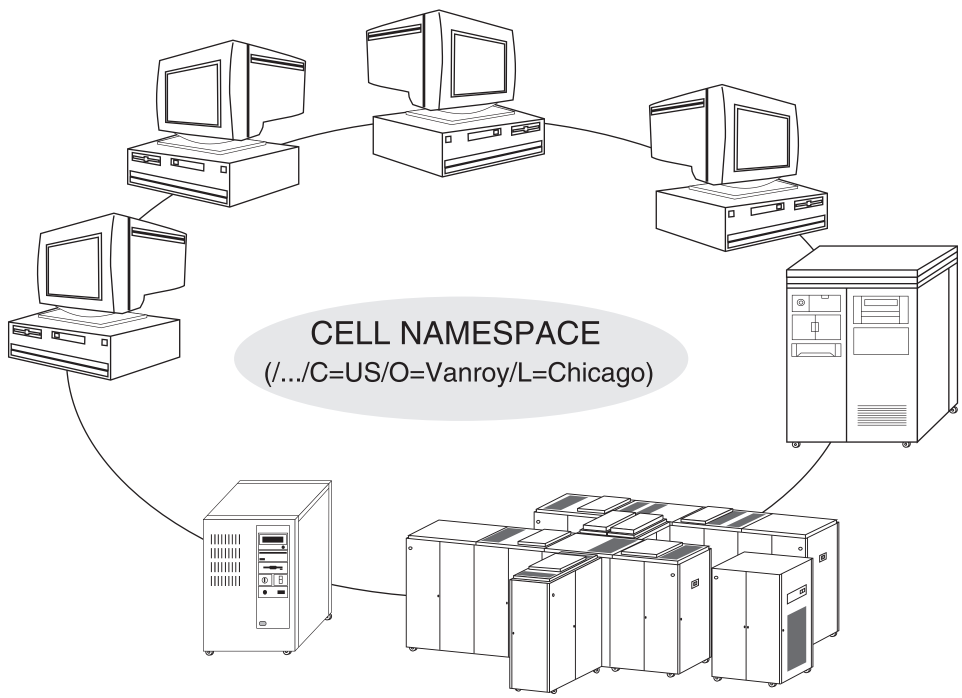

Figure 11. The Vanroy Cell

In a DNS implementation, Vanroy's Chicago domain name would be /.../chicago.vanroy.com.

In the United States, you can register your cell name with:

X.500

American National Standards Institute
11 West 42nd Street
New York, NY 10036

DNS

Government Systems, Inc.
Attention: Network Information Center
14200 Park Meadow Drive
Chantilly, Va 22021

Once you establish your organization name, you should set up a central administrative authority to manage names subordinate to your unique organization name. Refer to Figure 21 on page 42 for a detailed cell hierarchy.

Clients

A DCE client is a machine or a process that uses a specialized service during the course of its own work. A client can be a CDS clerk, acting as the intermediary between client applications and CDS servers. A client can also be DTS clerk, synchronizing its system time with a time server.

Because all DCE machines, including DCE server machines, can be DCE clients, Vanroy has configured all its systems to run the DCE client software. (See Figure 13 on page 28.) In this way, each workstation or host can request the DCE services of any DCE server. This makes the planning for the placement of servers a little easier, because the Vanroy planner knows that DCE services can be requested from anywhere in the cell.

Servers

A DCE server is a machine or process that provides a specialized service to DCE clients. DCE servers fit into three categories. (See Figure 12.)

- Servers for distributed computing (DCE)
- Servers exclusive to z/OS DCE
- Servers you create.

Servers for RPC, CDS, Security, DTS, and GDS are implemented by OSF DCE. Using these common servers, you can distribute applications throughout your network even if it comprises different operating systems and different hardware platforms.

The optional Application Support servers are unique IBM additions to DCE. They make the transaction and batch strengths of CICS and IMS available throughout your distributed environment. Your CICS and IMS access is extended to your workstations and non-CICS and non-IMS address spaces on the z/OS system.

You can exploit the DCE's orientation toward heterogeneous systems by creating your own distributed applications. This will involve the creation and installation of your own servers on hosts in your network. See the *z/OS DCE Application Development Guide: Introduction and Style* and *z/OS DCE Application Development Guide: Core Components* for information on developing your distributed applications.

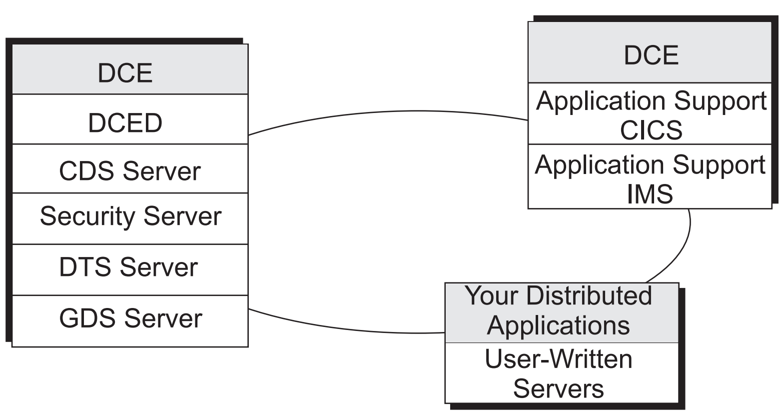

Figure 12. Types of DCE Servers

When a DCE server starts up, it registers its binding information based on the IP address of the host machine. If the server is installed on a host with multiple IP addresses, then multiple routes must be defined between clients and that server. If multiple routes are not defined, then clients may encounter problems when requesting a service from a server on that host. For example, a client that can only communicate with one of a host's IP addresses might select a connection from the server's binding information that it cannot access. In these situations an error will result. The network administrator must ensure that clients can connect to all network interfaces on a server. See the *z/OS DCE Application Development Guide: Core Components* for more information on network interfaces.

The following sections describe Vanroy's planning decisions for locating its DCE servers.

DCE Host Daemon

A DCE Host Daemon (DCED) must run on every machine that is going to be included in the cell, regardless of whether the machine is host or client. Clients require DCE Host Daemon to locate the requested process on the host server. CDS provides you with the address for a resource, while DCE Host Daemon locates the endpoint for the requested process. All Vanroy machines can request services (as DCE clients) and all Vanroy hosts can provide distributed services.

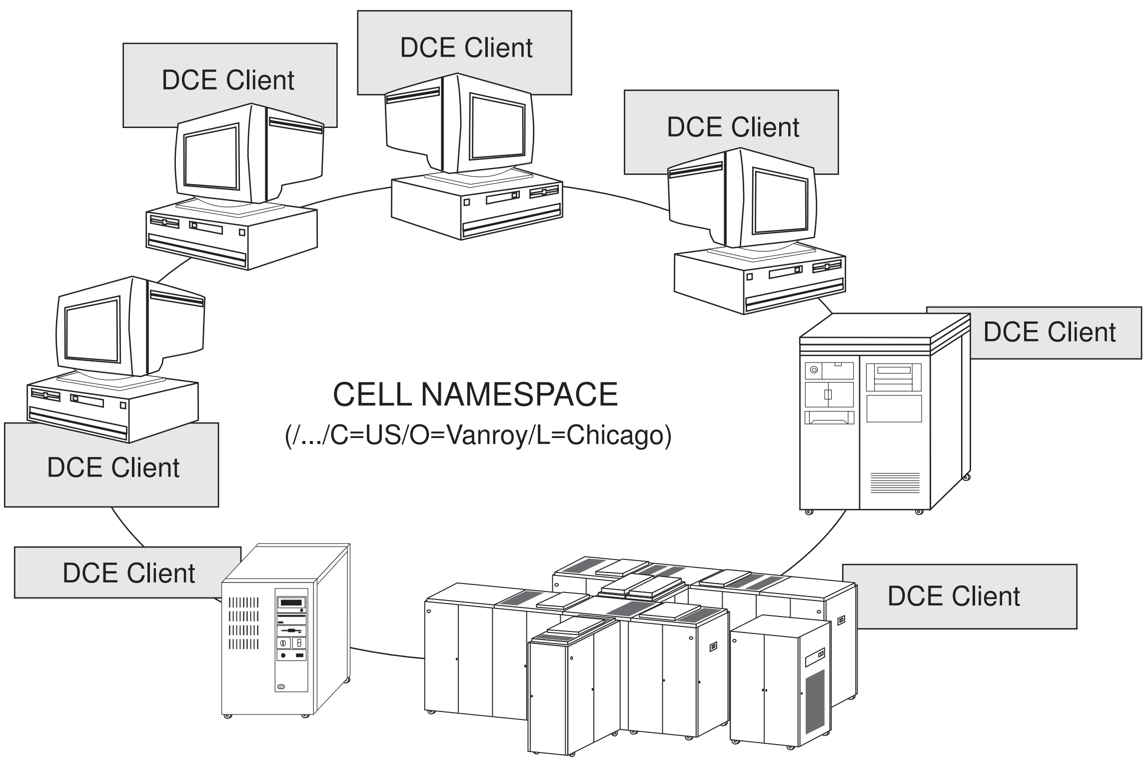

Figure 13. The Vanroy DCE Clients and DCE Host Daemons

Note: DCE Host Daemon must be running before you can configure any other z/OS DCE services, because their service endpoints must be registered with DCE Host Daemon.

CDS Server

CDS provides the method for naming and using a cell's resources. CDS lets you use resources anywhere in the cell without having to know their physical location. Given the unique name of a server or resource, CDS can return the address and other information associated with that name.

Every cell must contain at least one CDS server. It stores and maintains names and handles requests to create, modify, and look up data. The total collection of objects registered as names in a cell is called the cell namespace. The CDS names are typically organized into a hierarchical structure of directories. A CDS directory is a grouping of resource information, including names and attributes. Figure 14 illustrates Vanroy's decision to organize its CDS namespaces at the city level. This hierarchy shows that the Chicago cell namespace is comprised of the Finance, Sales, and Shipping directories. Directories organize groups of similar object entries; in this example, employee names.

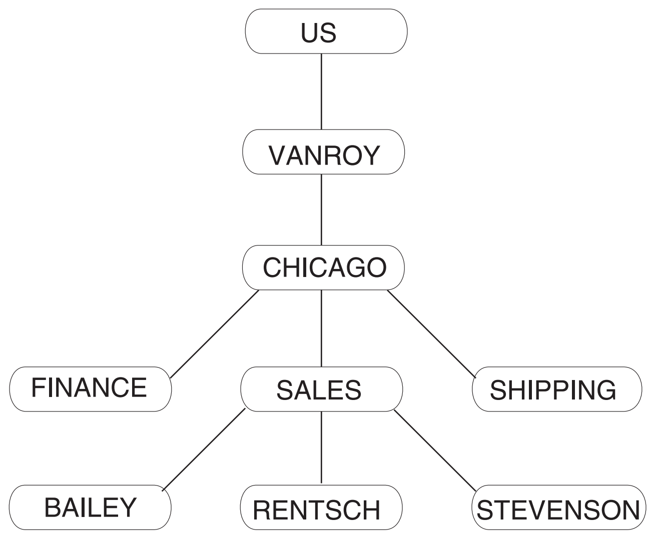

Figure 14. CDS Namespace

There can be more than one copy, or replica, of a given directory on other servers in the cell. When you create a replica of a directory, you replicate all of its entries as well. CDS ensures that the contents of all replicas of a directory are consistent. One copy of the directory is designated as the master replica. It is the first instance of a specific directory in the cell namespace. Any additional replicas are designated as read-only and are available for looking up information. Only the master replica can be modified directly.

The CDS advertiser receives and stores CDS server information and the status of those servers in the network. The CDS clerk serves as the intermediary between client applications and CDS servers. CDS advertiser and CDS clerk run on all client machines.

Because the directory service must be highly available and other services depend on it, the Vanroy Corporation decided to install two CDS servers. The master replica is installed on the AIX/6000 host and a read-only replica is located on the UNIX machine. As illustrated in Figure 15 on page 30, having the cell directory available on two different hosts gives Vanroy the most reliable way to back up namespace information and ensures continued DCE services should one of the hosts be unavailable.

Note: If you do not use z/OS for the CDS Server, z/OS DCE recommends that you use a channel attached AIX/6000 machine for the CDS master replica.

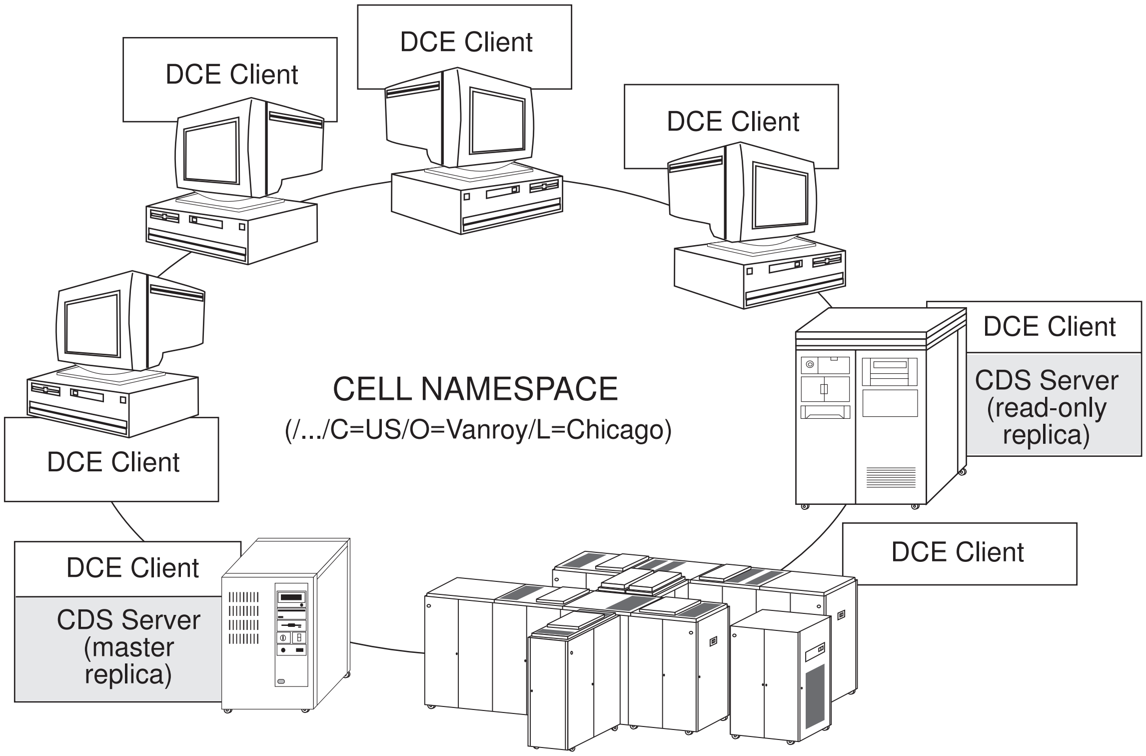

Figure 15. The Vanroy CDS Servers

When locating your CDS servers, keep the following guidelines in mind to achieve reliability, optimum performance, and data availability:

- Choose dependable nodes. A CDS server needs maximum availability with the ability to restart quickly. The CDS server needs to be one of the first systems available on the network, because clients and other DCE servers rely on it for up-to-date information.
- Use reliable network connections to ensure that all servers that maintain directory replicas can be reached when CDS performs periodic consistency checks.
- Consider the size of your cell and how geographically dispersed it is when deciding how many CDS servers you need. You need to have at least two copies of each directory, to ensure access to data should one of the servers be unavailable.
- You need to make replication decisions based on where the contents of directories are referenced. Put replicas on servers where the contents are mostly read, and put masters on servers where data is most likely written.

Local and Global Names: Whether your DCE environment is private, consisting of a single private cell or group of cells, or whether you interoperate with cells outside of your organization, you will use a hierarchical or tree structure to represent the names of your resources. A particular resource's name is the list of all the names associated with all the points on the path from that resource to the route of the tree. People, services, computers, storage devices, and printers can all be designated as resources.

The Vanroy Corporation is a single cell with its cell or CDS namespace boundary at the city level. Using the namespace illustrated in Figure 14 on page 29, the complete local or CDS name for salesperson Stevenson is /.:/Sales/Stevenson. Note that, in CDS, the cell root can be referred to either by the cell's root name (Chicago) or by the short form /.: prefix.

By contrast, in a global DCE environment, the GDS name for Stevenson might be /.../C=US/O=Vanroy/L=Chicago/Sales/Stevenson, while the DNS name could be /.../chicago.vanroy.com/sales/stevenson. The prefix /... indicates that a name is global. Chapter 5, "Additional Planning Considerations" on page 41 discusses using the GDS Server and global names for intercell communication.

Security Server

The Security Service maintains its own namespace within a cell's namespace. This ensures the trustworthiness of the security information in a DCE cell. Security Service clients use CDS to find the binding information that enables them to locate security servers. The node that runs the master security server must be highly available and physically secure.

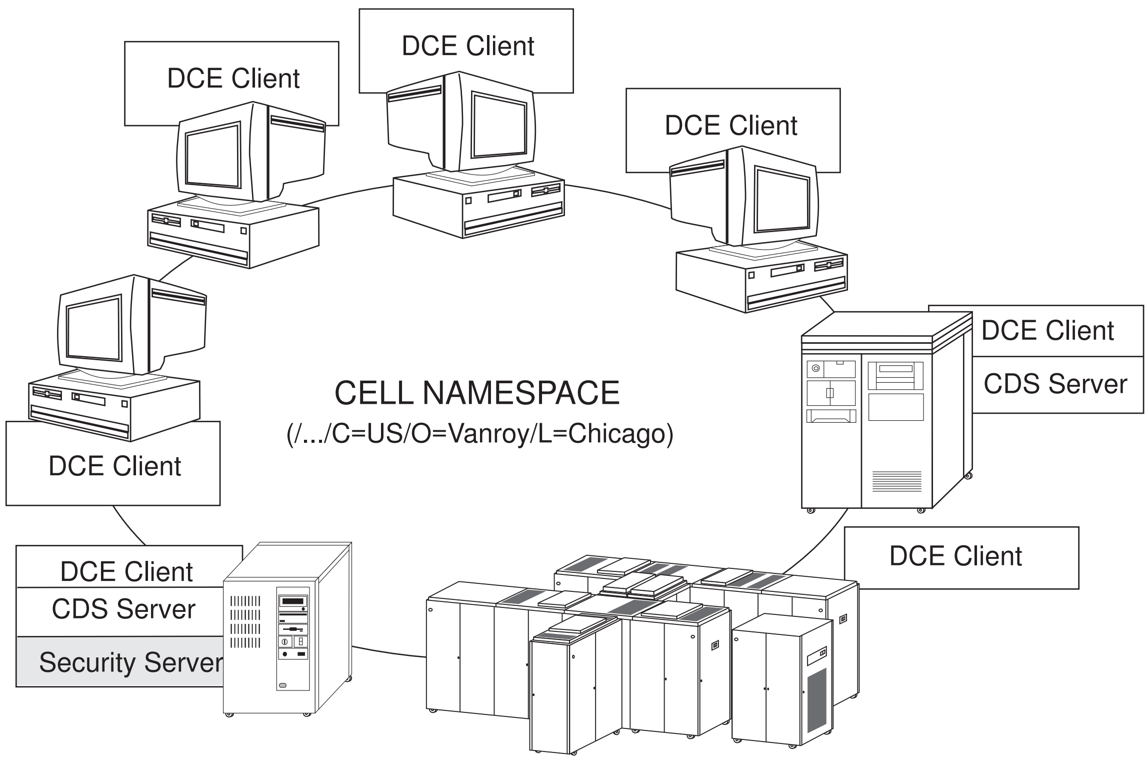

Figure 16. The Vanroy Security Server

You should configure the Security Server first, because every other server must authenticate with the Security Service.

Be sure to change the master registry site before removing the node from the network or shutting down the node for an extended period of time. If the master registry is unavailable, no security updates can be made.

A cell can have only one master registry. If you plan to merge several existing cells into a single cell, you must first merge their registries.

DTS Server

Each cell should have at least three DTS servers to arrive at a consensus of the correct time. A minimum of three are necessary to detect a faulty time server.

Vanroy has installed three DTS servers, one on each of its hosts. (See Figure 17 on page 32.) Remember that z/OS DCE time synchronizations do not adjust z/OS hardware clocks. As a result time obtained by z/OS services might be slightly different than time obtained by z/OS DCE services.

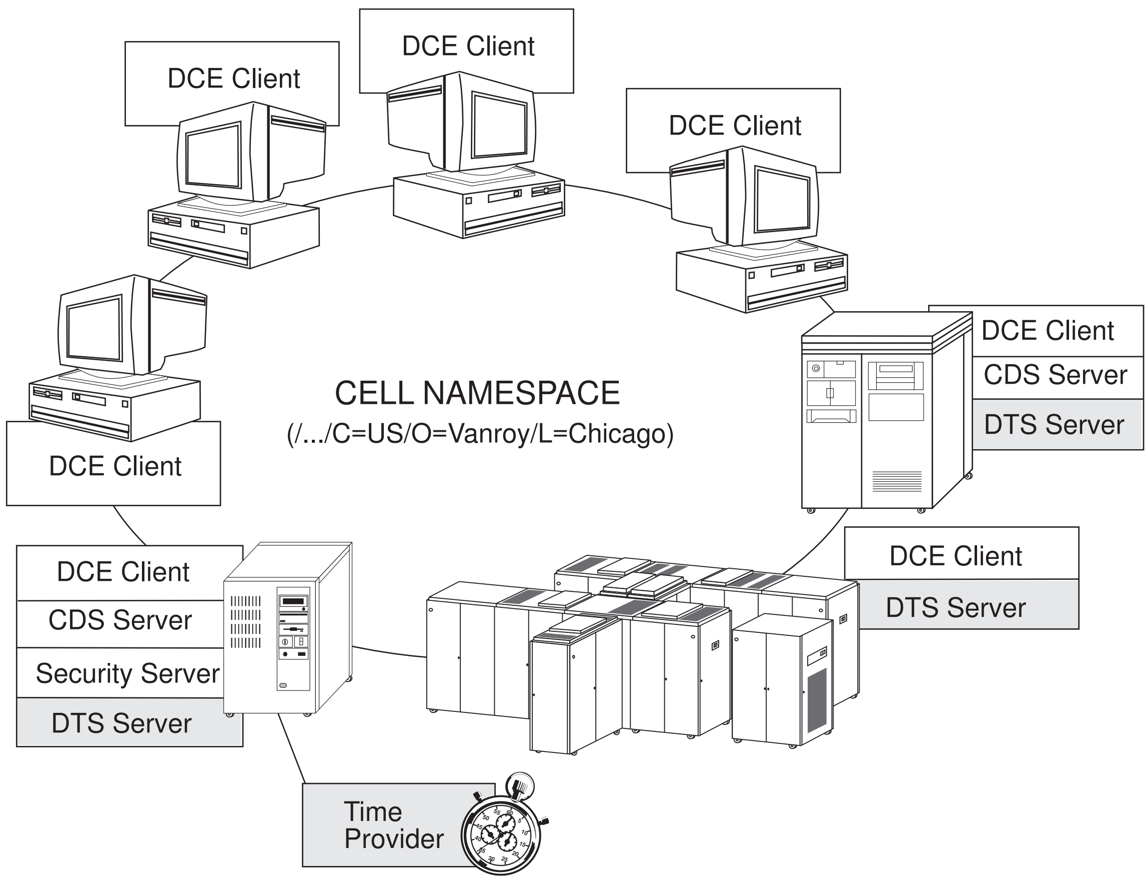

Figure 17. The Vanroy Distributed Time Servers

Four or more time servers provide redundancy. In this way, if a server fails, DTS can still synchronize with reliable results. A greater number of servers also increases the accuracy of time synchronization. However, remember that increasing the number of servers queried also increases the activity on the network and may affect performance.

Where it is not possible to configure three hosts as DTS servers, you can configure your z/OS machine as a DTS server and use the Null Time Provider (NTP) described in "Distributed Time Service" on page 14. Other hosts can then be configured as DTS clerks, using the z/OS machine for synchronizations.

IBM z/OS DCE Application Support

IBM z/OS DCE Application Support defines a single server for access to CICS or IMS data. This allows DCE RPC to exploit CICS and IMS transaction programs. Client programs running on programmable workstations or in non-CICS/non-IMS address spaces on z/OS (TSO or batch) can have transparent invocation of CICS and IMS programs. Vanroy has installed both an DCE AS CICS Server and an DCE AS IMS Application Support Server on its z/OS host. (See Figure 18 on page 34.) They provide the support necessary to extend the use of existing CICS and IMS transaction programs throughout its cell, and create an environment for the development of new CICS and IMS transaction programs as RPC server operations.

Application Support preserves the existing skills of Vanroy's z/OS COBOL programmers and workstation C programmers. Programmers can continue working in their areas of expertise, and produce client/server applications that work between z/OS and any number of clients. By isolating the application development from the interface, Application Support ensures that developer productivity is not lessened. The highly portable client code allows Vanroy to easily enable Application Support clients throughout its DCE environment.

Users of the Application Support server must be registered with both the z/OS security subsystem (for example, RACF), and the DCE security registry. A client must have a valid z/OS user ID to access the CICS and IMS regions and resources. A local z/OS account must be created for each Application Support server client. These clients also require a valid DCE user ID and account.

To gain access to CICS or IMS resources, the Application Support client is authenticated using the DCE Security Service, and authorization is validated through DCE ACLs. A mapping between a client's DCE ID and its z/OS identity is maintained, making access to CICS and IMS transparent to the Application Support client.

Before configuring your Application Support servers, you must:

- Install and configure z/OS DCE, including CDS and Security servers.
- Grant DCE permissions to the Application Support administrator. Usually, the DCE administrator is not familiar with CICS or IMS and will create accounts giving DCE permissions to an Application Support administrator.

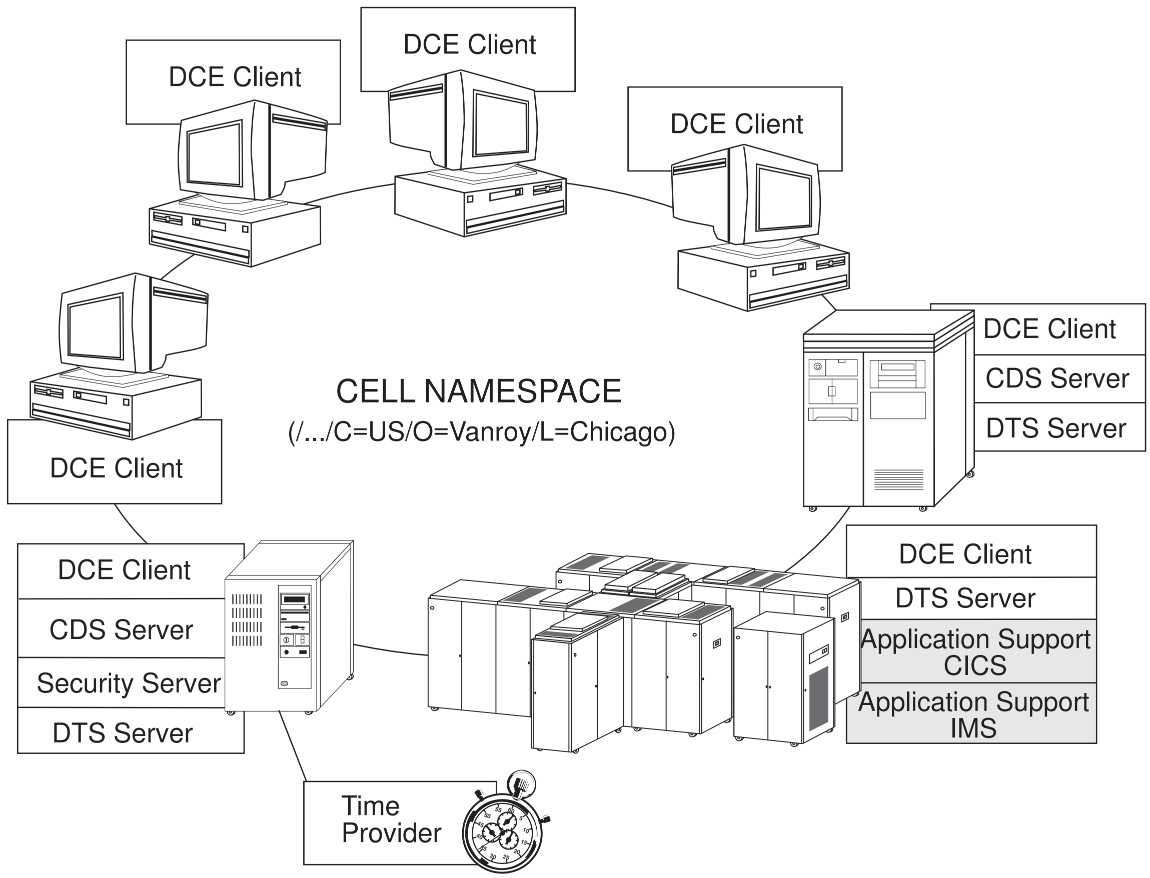

Figure 18. The Vanroy Application Support Servers

Application Support servers provide ISPF panels for administration of the servers.

Detailed information on using the DCE AS CICS and DCE AS IMS servers can be found in the *z/OS DCE Application Support Configuration and Administration Guide*.

User-Written Servers

Application Support server provides you with two servers that expand the transaction facilities CICS and IMS throughout your cell. You can also exploit the various computing strengths of your systems by developing your own application servers to run in DCE. You can extend functions to users who previously had no access to those features or resources. For example, your systems that perform rapid calculations, do detailed formatting, or print colored graphics can be made available to all users in your cell.

If your DCE application server is running on z/OS DCE, you may want to use the RACF Interoperability function, which allows the DCE application server to create the DCE client's corresponding z/OS credentials and use RACF for authentication. See *z/OS DCE Administration Guide*.

Figure 19 on page 36 shows that the Vanroy Corporation has developed its own user-written server to run on its z/OS host. As an example of server potential, the Vanroy server has been designed to give DB2® access to selected clients in the cell.

z/OS strengths that you might want to exploit through your own application servers include:

- Computational ability

- Vector processors
- I/O ability
  - 256 channels
  - Bandwidth
- Data storage capacity
  - DASD
  - Tapes
- System availability
  - Potential for 24 hours a day 7 days per week
- Hardware features
  - SYSPLEX
  - Printers
- z/OS associated products
  - DB2
  - VSAM
  - TSO
  - RACF
  - Batch facilities

Detailed information on the creation of user-written servers can be found in the *z/OS DCE Application Development Guide: Core Components*.

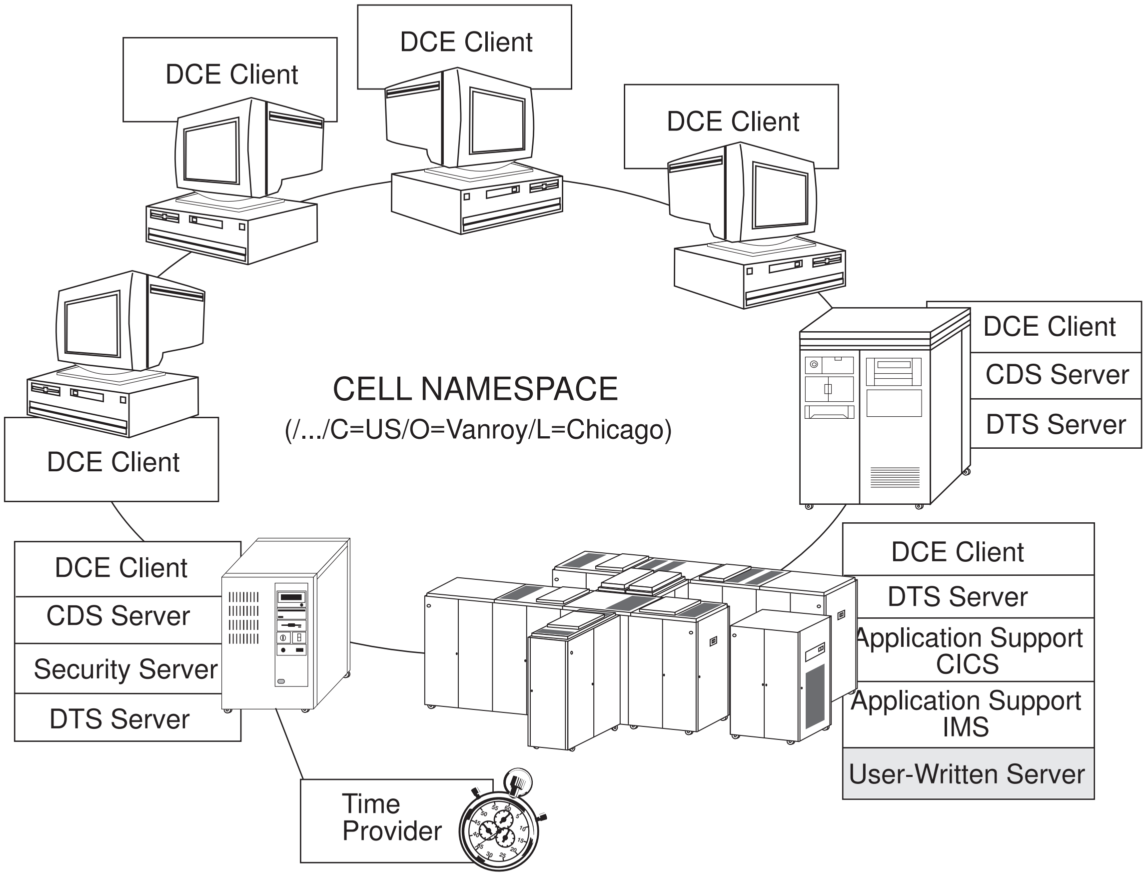

Figure 19. The Vanroy User Written Servers

Data Replication and Distribution

In addition to planning decisions regarding the placement of the various z/OS DCE components (clients, servers, daemons), you must make decisions regarding the placement and distribution of information in your DCE. Your decisions must answer these questions:

- How much information needs to be shared?
- Where should shared information be stored?
- How much information updating is required?
- How much information is for look up only?
- Is important data always available?

The following sections describe the considerations to keep in mind when you are planning the access to information in your cell.

Sharing: If a large volume of information needs to be shared, consider the amount of disk space you require. Also consider information that should be stored on a host and items that could be available on a workstation. If shared data is stored on a workstation, DCE Host Daemon must be installed on that workstation, so that clients can access that workstation information.

Because of substantial storage capacity of their z/OS system, Vanroy has decided to store most of its shared information on this host. The earlier decision to only install DCE Host Daemon on its hosts means that only private data will be stored on the workstations.

Updating: If information changes frequently and users rely on the accuracy of that information, you need to consider how much information you replicate. It is better to have a central source for information that changes frequently. If users look up shared information but do not need to change it, you can rely more on replicated data.

Vanroy has both information that is stable (employee telephone numbers) and information that changes frequently (financial data). Customer account information must always be current and therefore will only be available through CICS on the z/OS host. Because the listing of employee phone numbers is updated on a monthly basis, it can be replicated on another host.

Availability: Because other services depend on Security and parts of the CDS namespace, this data needs to be available at all times. Each CDS directory can be replicated, while the Security database can only be replicated in its entirety.

Vanroy replicates its CDS server on another host and installs its Security server on its most reliable node.

Keep in mind, that while replicating data helps availability, there is an associated cost in terms of performance and the amount of administration required. For example, replicated data must be synchronized with the master file.

Node Participation

z/OS DCE provides the DCE code necessary to allow your z/OS host to participate as both a client and server in your DCE cell. Your host can be a server for a subset of the DCE services, Application Support, and any user-written distributed applications. It can also be a client for the distributed services and applications available in your DCE environment.

The Vanroy z/OS host is active as both a client and a server in the Vanroy cell. As a client, it can employ user-written servers running on other nodes to exploit the strengths of those Vanroy systems. As a server, it provides the Vanroy cell with z/OS DCE services, the DCE AS CICS, the DCE AS IMS, and a user-written DB2 application.

For a machine to be an active participant in a DCE cell, it must have DCE for its operating system installed. The four Vanroy workstations, as DCE clients, must at the very minimum have the DCE client code for their operating system installed. The z/OS DCE code does not enable Vanroy workstations to be DCE clients.

The situation is similar for the other two hosts in the Vanroy cell. To be DCE clients, servers, or both, they must have DCE for their operating systems installed.

Figure 20 on page 38 shows that it takes four different DCE products (one per operating system) to provide DCE function to all nodes in the Vanroy cell.

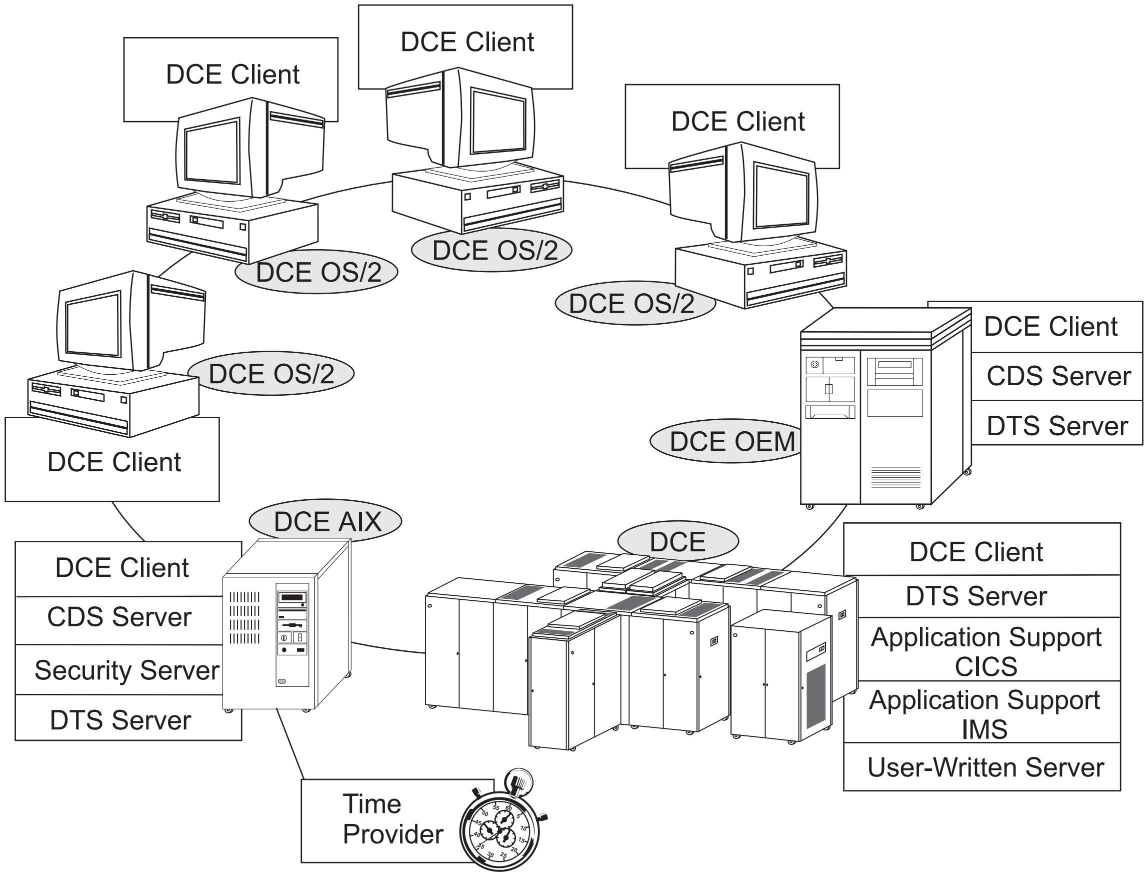

Figure 20. Vanroy Operating Systems Participation in DCE

Although the Vanroy scenario primarily involves IBM operating systems, DCE's heterogeneous orientation means that the Vanroy hardware and systems could be replaced with products from other computer companies. But regardless of the source of equipment, the rule is the same: to participate in a DCE environment, a node must have DCE for its operating system installed.

Node Communication: Nodes in your organization may already be communicating by means of any number of different network connections (for example VTAM®, token-ring). To communicate in a DCE environment, your machines must be connected by the Transmission Control Protocol/Internet Protocol (TCP/IP).

Planning Checklist

This checklist summarizes the planning questions asked when the Vanroy Corporation designed its distributed computing environment. You should consider these same questions when planning your z/OS DCE environment.

___ How big should your cell be (common purpose and trust)?

___ Do you need more than one cell?

 ___ How much information is shared?
 ___ How much information is restricted?

___ Does your cell include more than one LAN?

___ Will you be communicating with cells outside of your organization (now or in the future)?

   ___ Will you be X.500 enabled?
   ___ Will you be an Internet site?

___ Who will manage the names in your cell?

___ Who will be the clients for z/OS DCE?

   ___ Should all systems be enabled as clients?

___ Which hosts will run RPC-based servers?

   ___ Should all hosts (with or without servers) run DCE Host Daemon?

___ Where will the master directory exist?

   ___ Is this node highly available?

___ Will you replicate the directory on another CDS server?

___ Where will the master security registry exist?

   ___ Is this node highly available?
   ___ Will you replicate this registry?

___ Do you need RACF-DCE interoperability? You do if the answer to any of the following questions is “yes”:

   ___ Do you plan to write DCE servers that require RACF for authorization to z/OS resources?
   ___ Do you want your z/OS users to be able to authenticate themselves to z/OS and then run DCE applications without having to reauthenticate to DCE?

___ Where will you install DTS servers (recommended three)?

___ Will you use an external time-provider as a UTC time source?

___ Do you have CICS or IMS installed on your z/OS host?

   ___ Do you want access to CICS or IMS applications available throughout the cell?

___ Will you create your own distributed applications?

   ___ Where will you install your user-written servers?

___ Where will shared information be stored?

___ How will you insure information accuracy?

___ How will you insure information availability?

___ Do you have DCE for all operating systems participating in your cell?

Chapter 5. Additional Planning Considerations

The previous chapter described a sample planning scenario for a single celled DCE implementation. Although a simple example, the Vanroy Corporation scenario posed many questions that all DCE planners must address regardless of the complexity of their environment. This chapter builds on the Vanroy scenario, adding some planning variables that might apply to you when you are planning your DCE implementation.

Multiple Cells

If you think your environment will grow rapidly, you might consider setting up several cells representing small areas of your organization. It will be easier to manage these smaller units as your network expands. Members of each cell share a common purpose and the cell is a unit of administration and security. If you anticipate slow expansion of your network, you may only need to establish one cell based on your current organization. You need to consider how many administrators you will need to maintain your cells, based on anticipated future growth.

If you choose to establish a multicelled environment, or if you will be interoperating with other cells in the X.500 namespace, or if you will be working with resources that are not associated with a cell, you will need to install the Global Directory Service (GDS).

CDS and GDS Names: GDS acts as a high-level connector allowing resources to find and interact with one another. These resources fall into categories: those existing in a global namespace and those belonging to a cell. Resources not belonging to a cell exist in the global namespace, while members of a cell make up the cell namespace. The structure of global names for each category is slightly different. This section compares the two name styles. Cell independent resources, that is those resources not belonging to a cell, will be referred to as pure GDS-style names.

The GDS database is a hierarchical, object-oriented database. Information exists in the global directory, similar to a CDS namespace, called a Directory Information Tree (DIT). Every GDS entry has a distinguished name, which uniquely and unambiguously identifies that entry. The distinguished name consists of a sequence of valid relative distinguished names (RDNs). Each RDN consists of a combination of the type and value of an attribute at a particular position in the DIT. Attribute types indicate the nature of the information stored in the attribute value.

Figure 21 on page 42 illustrates a path through the hierarchy of attribute types in searching for a Chicago-based salesperson named Stevenson. In this example, salesperson Stevenson exists in the global namespace, and does not belong to a cell. The scenario would be the same for hardware or services in the global namespace.

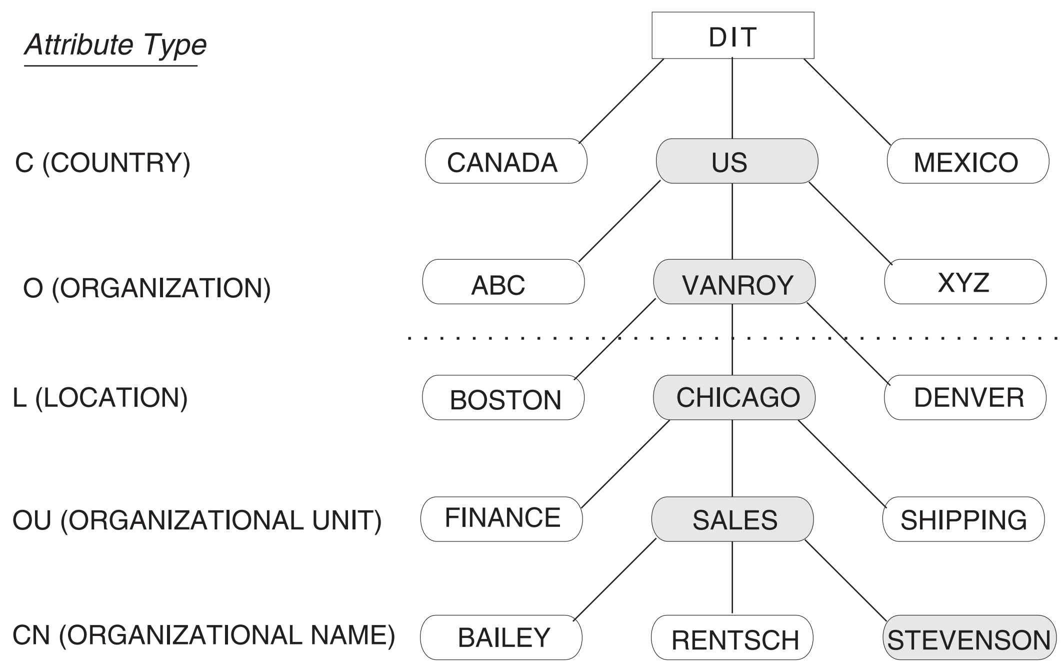

Figure 21. Directory Information Tree

Table 1 shows the RDNs and resulting distinguished name for Chicago salesperson Stevenson. The table also shows the global convention for distinguished names. Each distinguished name starts with a representation of the global root (/...). Attribute types and values are separated by an equal sign (=), and each RDN is separated by a slash (/). Each attribute value that makes up an RDN is called a distinguished value.

Table 1. Distinguished Name

Attribute Type	Attribute Value	Relative Distinguished Name	Distinguished Name
C	US	C=US	/.../C=US
O	Vanroy	O=Vanroy	/.../C=US/O=Vanroy
L	Chicago	L=Chicago	/.../C=US/O=Vanroy/L=Chicago
OU	Sales	OU=Sales	/.../C=US/O=Vanroy/L=Chicago/OU=Sales
CN	Stevenson	CN=Stevenson	/.../C=US/O=Vanroy/L=Chicago/OU=Sales/CN=Stevenson

The structure of the GDS name points out an important difference between GDS and CDS. A CDS name is completely distinct from its attributes. It consists of a string of directory names ending with the simple name of the entry being referenced. In contrast, a GDS name consists of a series of attribute types and their values. Using the example in Figure 21, the GDS distinguished name is /.../C=US/O=Vanroy/L=Chicago/Group=Sales/Name=Stevenson, which is a concatenation of attribute types and values, one from each level of the DIT. If the CDS namespace boundary is at the city level, then /Sales/Stevenson is the directory specification for salesperson Stevenson. Attributes and their values are not part of the CDS full name.

Global names for entries belonging to cells are slightly different from pure GDS-style names. The global names for entries created in CDS consist of two parts: the first portion is the global cell name, with the balance being the CDS name. Because salesperson Stevenson is a member of the Chicago cell, the resulting global name is /.../C=US/O=Vanroy/L=Chicago/Sales/Stevenson.

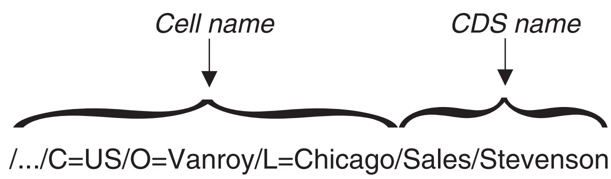

Figure 22. Global Name for a Cell Member

In the global namespace, the prefix /... represents the global root. In CDS, the cell root can be referred to either by the cell's global name (Chicago) or by the short form /.: prefix.

CDS and DNS Names: The DNS or Internet name space is also a system of hierarchically organized names. The DNS tree is anchored at the root and then divided into several top-level domains. Top-level domains were originally based on type of organization (explained in "Unique DNS Cell Names" on page 25), but two-letter abbreviations reserved for each country have been added to this level. A particular node's domain name is a list of labels associated with all the nodes on the path from that node to the root of the tree.

Figure 23 illustrates the path through the Internet nodes to Vanroy's Chicago domain. The break in the tree between the city level and the department level indicates that departments and their personnel are members of a city's domain and not nodes on the DNS tree.

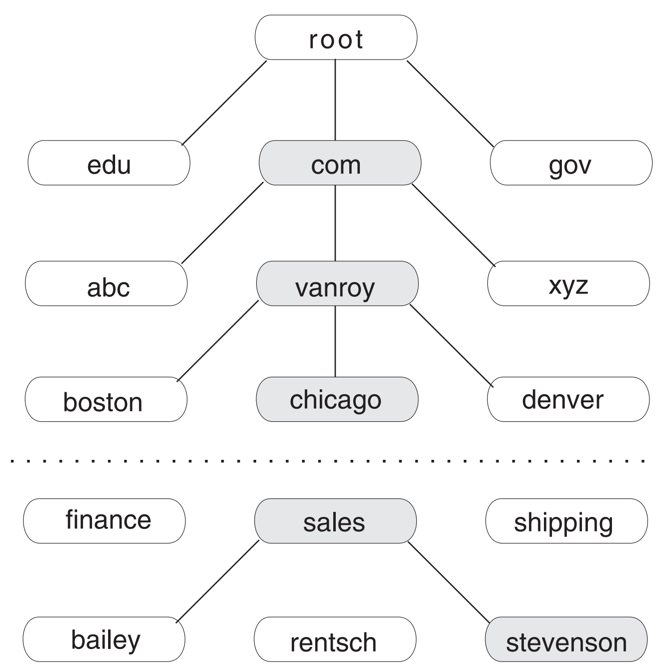

Figure 23. A Domain Tree

The resulting DNS name for Chicago salesperson Stevenson is /.../chicago.vanroy.com/sales/stevenson. The DNS domain name is the cell name: /.../chicago.vanroy.com. The remainder of the name is CDS name sales/stevenson.

GDS Server

Like CDS, the global directory must be highly available. As discussed in "Global Directory Service" on page 13, GDS and its associated GDA are required by DCE for global name resolution.

In the Vanroy cell scenario, it was decided that the AIX/6000 host was a reliable node and so the master replica for the cell directory was stored there. Figure 24 on page 44 shows an expanded Vanroy Corporation, with a new cell located in Boston. By installing GDS in the Chicago cell, DCE clients in Chicago now have access to a z/OS DCE server in Boston.

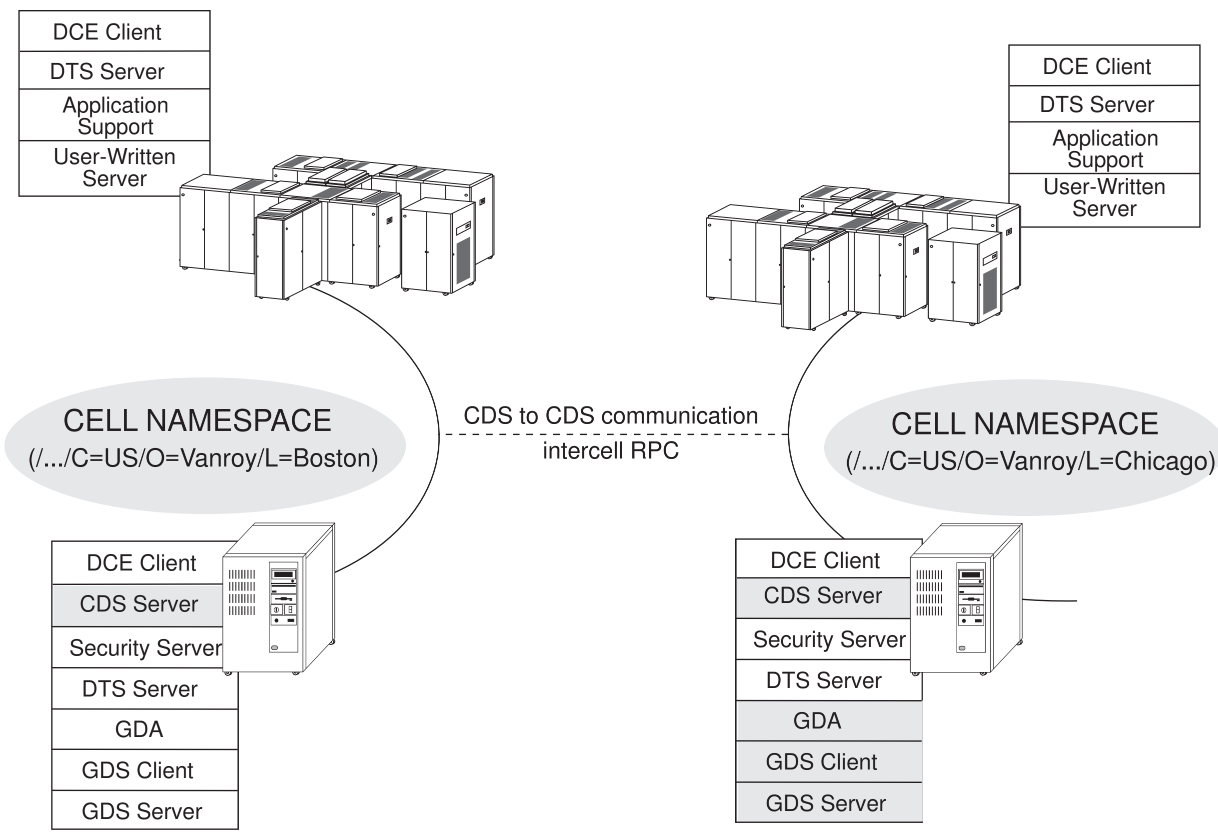

Figure 24. A Multicell Environment

Figure 25 on page 45 provides a simplified high-level overview of global name resolution. The CDS clerk in the Chicago cell first searches its cache for a previous response to a similar query. If the information is not in the cache, the CDS clerk checks the cell name. If the clerk identifies the request as intercell (requested cell name different than the local cell name), it is sent to the GDA. By contrast, a local request is sent to the CDS server for resolution. Because Vanroy names are typed (X.500 names), GDA name resolution initiates a call that requests resource-attribute information from the GDS server. The resource address is returned to the CDS clerk, allowing an RPC call to the Boston server.

GDA imitates a CDS server; it looks for a cell entry for the requested resource in either GDS or DNS, depending on the format of the resource name. GDS names are typed, consisting of a type and a value separated by an equal sign (=). DNS names are untyped, consisting only of values. Figure 25 illustrates GDA name resolution, for typed names (/C=US/O=Vanroy/L=Boston) and for untyped names (vanroy.boston.com).

All this activity among GDA and GDS server takes place on the AIX® host.

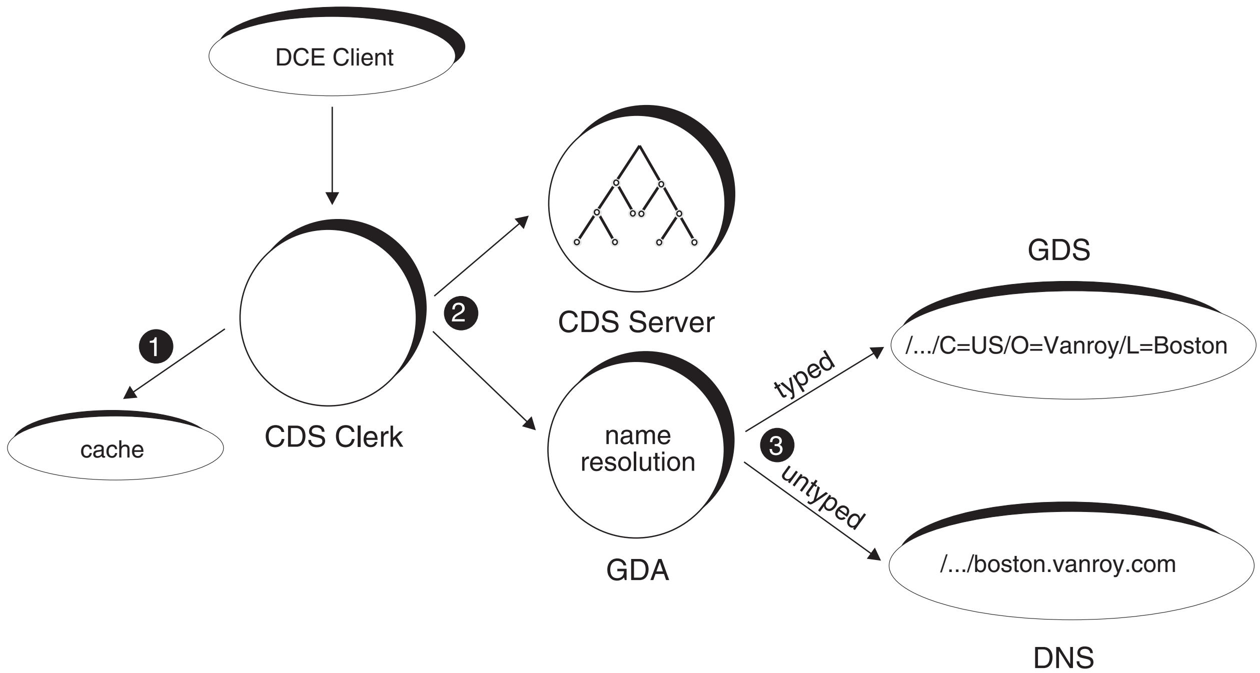

Figure 25. Directory Name Resolution

DTS in LANs and WANs

If your cell nodes are in a single LAN, as in "A Sample Cell" on page 23, your DTS implementation is relatively simple. It is recommended that you configure a minimum of three systems as DTS servers. An external time provider is recommended if you need to synchronize with UTC.

Extended LANs

If your cell is made up of several LANs, as in Figure 26 on page 46, for best performance you should create three DTS servers in each LAN. One server in each LAN should be configured as a global time server. One server in each LAN should be configured as a courier time server, for synchronization with the global server. Time servers and clerks are described in "Distributed Time Service" on page 14. Courier servers will request time values from one randomly selected global server at every synchronization to enable you to have a shared sense of time throughout your cell. The global time server and the courier time server can be installed on the same node.

If you are using external time providers, connect them to your global servers.

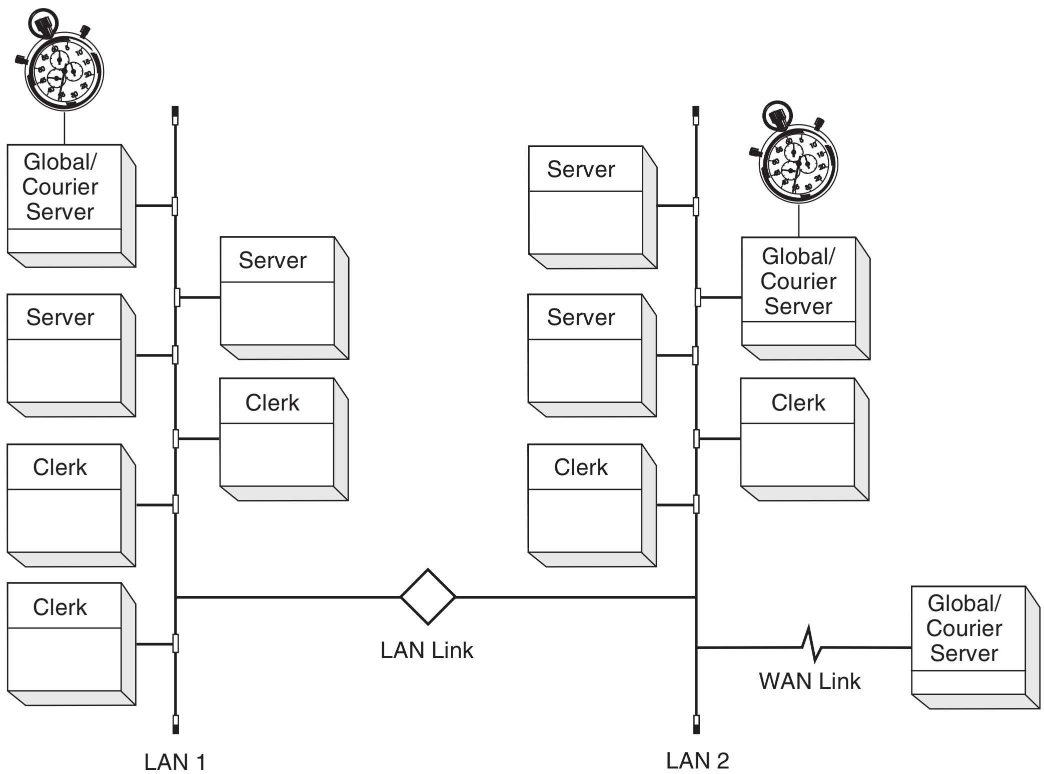

Figure 26. DTS Configurations in LANs and WANs

WANs

Because there are numerous variations of WANs, especially in combination with LANs and extended LANs, this section provides only a few generic configuration examples.

LANs with WAN Links to Remote Sites: Figure 26 shows a WAN link between LAN 2 and a remote node. In this configuration, you should:

- Configure servers at remote nodes as global and courier servers.
- Have a minimum of three servers on the LAN.
- If using a single time provider, locate it on a global server on the LAN.

In this configuration, DTS is not dependent on remote nodes for synchronization.

LANs Connected by WAN Links: The configuration suggestions for extended LANs also apply to LANs connected by WAN links.

WAN Cells: Even in a geographically distributed cell that does not have any LANs, DTS can still provide adequate synchronization for most distributed applications. Keep the following in mind when planning an all WAN cell:

- You should have at least three servers.
- Configure all servers as global servers.
- Place at least one server at the central site.
- Place remote severs at nodes connected by the most reliable communication links.

- Connect a time provider to a global server at the central site.

Multiple Application Servers

Application Support defines a single server for access to CICS or IMS data. If additional throughput is required, additional Application Support servers can be started. Each Application Support server must be defined to CDS as a separate server with different unique identifier. Clients can select interfaces from a specific Application Support server by specifying its unique identifier or can allow RPC to choose from the valid candidates.

Slow throughput can result from many simultaneous client requests for Application Support servers. You should consider starting an additional Application Support server.

Chapter 6. Planning For Installation

This chapter outlines the steps to follow when you are planning for the installation of z/OS DCE.

If you are migrating from OS/390® Release 7 or later, there are no migration actions for z/OS. If you are migrating to z/OS from a release earlier than OS/390 Release 7, follow the migration actions specified in the appropriate section that follows.

Migrating to z/OS from OS/390 Release 6 or Earlier

For z/OS:

- The security registry database consists of the HFS files in the **opt/dcelocal/var/security/rgy_data** directory or the DB2 SRGYDATA database. The DCE security registry database in z/OS is incompatible with releases of OS/390 earlier than Release 7. Therefore, ***it is strongly recommended that you back up the registry database before bringing up DCE at the z/OS level*** in case you want to revert back to an earlier version of the security server. The first time that the z/OS DCE security server is started, it migrates an existing registry database to the new format. After this has been done, the registry database cannot be used with an earlier version of the security server.
- Client workstations that run a version of DCE that is based on OSF DCE version 1.2.2, are able to interoperate with z/OS DCE. OSF DCE was developed by the Open Software Foundation (OSF), renamed The Open Group.
- If you wish to change environment variable (**envar**) files, do not make changes to the default environment variable files in **/usr/lpp/dce/home/**, or you will lose your changes when you upgrade to the next release. For z/OS DCE all of the default environment variable files are shipped in **/usr/lpp/dce/home/**. Symbolic links (or *symlinks*) in **/etc/dce/home/** point to these environment variable files. In addition, on z/OS DCE, **/opt/dcelocal/home** is a link to **/etc/dce/home**.

  Note: In z/OS DCE, use code page IBM-1047 when updating environment variable files.

  To change the environment variable files:

  1. Rename the environment variable symlink.
  2. Copy the default file from **/usr/lpp/dce/home/*xxx*/envar** to **/etc/dce/home/*xxx*/envar** (*xxx* is a specific daemon). For example:

     ```
     >>cd/etc/dce/home/dced
     /etc/dce/home/decd>> mv envar envarold
     /etc/dce/home/dced>> cp envarold envar
     ```

  3. Make your updates to the file just created in **/etc/**.

There are a number of additional migration considerations when you are migrating from a release of OS/390 earlier than Release 5. These are described in the remainder of this section.

If you are skipping one or more releases (for example, if you are migrating from OS/390 DCE Release 1 to z/OS you may find this table helpful:

Table 2 (Page 1 of 2). Migrating to z/OS DCE from an Earlier Release

Migrating from	What to Do
MVS 5.2.2 or MVS 5.1	Start at "Migrating to z/OS from MVS 5.2.2 or MVS 5.1" on page 50 and then do the steps for the later releases.

Table 2 (Page 2 of 2). Migrating to z/OS DCE from an Earlier Release

Migrating from	What to Do
OS/390 Release 1, 2, or 3	Start at "Migrating to z/OS from OS/390 DCE Releases 1, 2, or 3" on page 50 and then do the steps for the later releases.
OS/390 Release 4	Start at "Migrating to z/OS from OS/390 DCE Release 4 or Earlier" on page 52 and then do the steps for the later releases.
OS/390 Release 5 or later	No action required.

Migrating to z/OS from MVS 5.2.2 or MVS 5.1

If you are migrating from MVS 5.2.2 or MVS 5.1 directly to z/OS DCE without upgrading to an intermediate release, there are several steps that you must complete:

1. Save the endpoint map access control lists (ACLs)

   Run the sample job EUVSVACL to save the ACL from your old release of DCE Base Services (at MVS/ESA™ 5.*x* level). The sample job EUVSVACL is found in *prefix*.HMB3160.F1. The job creates a file called **/tmp/migrate.script** that contains **dcecp** commands to migrate the existing ACLs for a DCE Base Services RPCD endpoint map to ACLs associated with a DCE Base Services DCED endpoint map. EUVSVACL must be run before DCE Base Services is configured.

2. Deconfigure your old DCE Base Services.

   If you are installing your new DCE Base Services on the same host machine and DCE cell as your old DCE Base Services, you must deconfigure the old DCE Base Services from the cell before configuring your new DCE Base Services. See *z/OS DCE Configuring and Getting Started* for more information on deconfiguring.

   Note: Applications that run in the DCE environment may be affected by deconfiguring. See the documentation for those applications to determine whether any other actions are necessary prior to the DCE deconfiguration. For example, DCE Distributed File Service should be deconfigured prior to deconfiguring the DCE Base.

   If you are using CBPDO to install DCE, these deconfiguration steps are completed during Wave 2 of installation, as described in the section on pre-APPLY migration actions in the *z/OS Program Directory*.

3. Restore endpoint map ACLs.

   After you have completed the configuration of DCE Base Services, run the **/tmp/migrate.script** shell script to restore the ACLs associated with the old DCE Base Services RPCD endpoint map.

Migrating to z/OS from OS/390 DCE Releases 1, 2, or 3

If you are migrating from OS/390 Release 1, 2, or 3, there have been changes that affect DCE function and actions that you should know about, because they may require changes in the way you install z/OS DCE and use it in your applications.

Installation Considerations: Modifications to the installation procedures consist of changes to:

- Changes to files shipped with DCE
  - Message catalogs and files
  - HFS files
- Changes to environment variables, procs, and data definitions

In addition, some changes to applications may be required.

Changes to Files Shipped with DCE – Message Catalogs and Files: Because z/OS DCE uses the z/OS UNIX System Services **gencat** facility, the format of message catalogs differs from that in OS/390 DCE Releases 1-3, and the contents of **/usr/lib/nls/msg** have been changed and expanded. You must ensure that the new structure and catalogs are in place prior to starting DCEKERN. This is handled as part of the installation processing. If you must migrate back to OS/390 DCE Release 1, 2, or 3, make sure that the file systems with the appropriate (earlier release) message catalogs are active when you are using the earlier OS/390 DCE.

Changes to Files Shipped with DCE – HFS Files: Due to the availability of a Cell Directory server and Global Directory Agent, in z/OS DCE the Daemon Configuration File (HFS file **/opt/dcelocal/etc/euvpdcf**) differs from that of OS/390 DCE Release 1-3. The line for EUVCDSD has been activated and a new line for EUVGDAD has been added. See the "Daemon Configuration File" topic in *z/OS DCE Configuring and Getting Started* for an example of these lines. This file should be backed up (that is, a copy should be made) before you install the newer release. When migrating back you should copy over the appropriate **euvpdcf** file before starting DCEKERN.

If you choose to run the Cell Directory server, on z/OS DCE it is recommended that you create a separate file system for it at path **/opt/dcelocal/var/directory/cds** to hold the files it will create and manage. It is also recommended that you reserve 30 megabytes of space in **/opt/dcelocal/var/directory/cds** to hold the clearinghouse for the CDS server; or, if you do not create a new file system, that **/opt/dcelocal/var/dce** be increased by 30 megabytes.

Changes to Environment Variables, Procs, and Data Definitions: DCE daemons **cdsd** and **secd** default to run in their own address spaces rather than within the DCEKERN address space. The environment variable **_EUV_DAEMONS_IN_AS** can be specified in DCEKERN's environment variable file to override this default. If it is not set, the default action will occur. That is, **secd** and **cdsd**, if configured, will run in separate address spaces. If you are migrating back to the earlier release of OS/390 DCE, the environment variable will be ignored. Furthermore, if **cdsd** is configured, it will not be started. Also, **secd** will run inside DCEKERN.

See the *z/OS DCE Administration Guide* for more information on setting the **_EUV_DAEMONS_IN_AS** variable, and on which daemons can be started in separate address spaces.

There is a separate procedure for each daemon that is "split," that is, runs in a separate address space. The procedure is to be used *only* when a daemon is started in a separate address space. Otherwise, you should use the DCEKERN procedure, as usual. If you made any changes to DCEKERN in your earlier release (such as added STEPLIBs or a changed location of output) for a daemon that is split, the same changes must be made to the individual procedure for that daemon. The procedure names for the daemons that existed in your earlier release are DCEAUDD for **auditd**, DCEDTSTP for **dtstp**, DCEPWDD for **pwdmgmt**, and DCESECD for **secd**.

There is an environment variable, **_EUV_CDSD_FILESPACE_THRESHOLD**, for filespace threshold checking for **cdsd**. This variable, if specified, is set in the **envar** file for **cdsd**, which is in **/opt/dcelocal/home/cdsd/envar**. See the *z/OS DCE Administration Guide* for more information.

If you are using Serverpak to install DCE, do these steps after installation:

1. Create these directories in the **/etc** file system:
   a. /etc/dce/home/cdsd
   b. /etc/dce/home/gdad
2. Create these symbolic links:
   a. **ln -s /etc/dce/home/gdad/envar../../../../usr/lpp/dce/home/gdad/envar**
   b. **ln -s /etc/dce/home/cdsd/envar../../../../usr/lpp/dce/home/cdsd/envar**

If you are using CBPDO to install DCE, these steps are completed during Wave 2 of installation, as described in the section on HFS directories in the *z/OS Program Directory*.

Changes to Applications That May be Required: Starting with OS/390 Release 4, the EUVPDLL part is a C/C++ DLL, which is not compatible with earlier releases. Existing applications will work with this DLL format.

For any **new** applications, IBM strongly recommends that you compile all modules with the additional compile-time option `–W0,DLL`. Furthermore, the link-edit step must specify `–l dce EUVPDLL.x`.

Migrating to z/OS from OS/390 DCE Release 4 or Earlier

In OS/390 DCE Release 4, the local configuration data was stored primarily under the **/etc/dce/** directory structure, but there was data stored in **/krb5**, too. Starting with OS/390 DCE Release 5, all local configuration data is stored under the **/etc/dce/** directory structure. The data that was kept in **/krb5** is under a directory at **/etc/dce/var/krb5**, and there is a symbolic link from **/krb5** to **/etc/dce/var/krb5**.

To migrate a configured system from OS/390 Release 4 or earlier create an **/etc/dce/var/krb5** directory, and move the data from the **/krb5** directory into it. Then, create a symbolic link from **/krb5** to **/etc/dce/var/krb5**.

If you are using CBPDO to install DCE, these **/krb5** steps are completed during Wave 2 of installation, as described in the section on pre-APPLY migration actions in the *z/OS Program Directory*.

An OS/390 Release 4 configuration will continue to work with this directory structure. When you are ready to migrate, the **/etc/dce/** directory structure can be carried forward and mounted at the **/etc/dce/** mount point to preserve the configuration data.

Mandatory Preinstallation Planning

This section describes what you must do before installing the z/OS DCE. Use the *z/OS Program Directory* for instructions when you are installing the product. This chapter only gives a simple overview of the process.

Prerequisite Hardware and Software

Ensure that you have the appropriate hardware and software for z/OS DCE. See the manual, *z/OS Planning for Installation*, GA22-7504, for hardware and software requirements. For the latest information right before you install z/OS DCE, check the *z/OS Program Directory*.

Storage Requirements

Ensure that you have sufficient storage for z/OS DCE. See the manual, *z/OS Planning for Installation*, GA22-7504, for information on determining storage requirements. For the latest information right before installing, check the *z/OS Program Directory* that comes with the product.

System Modification Program/Extended (SMP/E)

Use SMP/E to install z/OS DCE and apply maintenance to it. SMP/E must be installed, and you must be familiar with its use.

z/OS DCE is installed using the SMP/E RECEIVE, APPLY, and ACCEPT commands.

Installation Decisions

z/OS DCE is distributed on either 9-track magnetic tapes, written at 6250 BPI, or on a 3480 cartridge, or 3480 IDRC, or 4mm cartridge. The tapes or cartridge contain all the programs and data necessary for installation.

Prior to installing z/OS DCE, read *z/OS Planning for Installation*, GA22-7504, and the *z/OS Program Directory*. The *z/OS Program Directory* tells you where to find any updates to the information and procedures presented in the z/OS DCE documentation.

Verification

The installation verification procedure (IVP) should be run to ensure that the installation process was completed successfully. Refer to the *z/OS Program Directory* for the z/OS DCE requirements when running IVP.

Appendix A. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

AIX	AIX/6000	BookManager
CICS	CICS/ESA	DB2
IBM	IBMLink	IMS
IMS/ESA	Library Reader	MVS/ESA
OS/390	RACF	Resource Link
SecureWay	VTAM	z/OS

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

This glossary defines technical terms and abbreviations used in z/OS DCE documentation. If you do not find the term you are looking for, refer to the index of the appropriate z/OS DCE manual or view the *IBM Glossary of Computing Terms*, located at:

http://www.ibm.com/ibm/terminology

This glossary includes terms and definitions from:

- *IBM Dictionary of Computing*, SC20-1699.
- *Information Technology—Portable Operating System Interface (POSIX),* from the POSIX series of standards for applications and user interfaces to open systems, copyrighted by the Institute of Electrical and Electronics Engineers (IEEE).
- *American National Standard Dictionary for Information Systems,* ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- *Information Technology Vocabulary,* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1.SC1).
- *CCITT Sixth Plenary Assembly Orange Book, Terms and Definitions* and working documents published by the International Telecommunication Union, Geneva, 1978.
- Open Software Foundation (OSF).

The following abbreviations indicate terms that are related to a particular DCE service:

CDS	Cell Directory Service
CICS/ESA®	Customer Information Control System/ESA
DTS	Distributed Time Service
GDS	Global Directory Service
IMS/ESA®	Information Management System/ESA
RPC	Remote Procedure Call
Security	Security Service
Threads	Threads Service
XDS	X/Open Directory Services
XOM	X/Open OSI-Abstract-Data Manipulation

A

access control list (ACL). (1) GDS: Specifies the users with their access rights to an object. (2) Security: Data that controls access to a protected object. An ACL specifies the privilege attributes needed to access the object and the permissions that may be granted, to the protected object, to principals that possess such privilege attributes.

access control list facility. A Security Service feature that checks a principal's access to an object. This facility determines access rights by comparing the principal's privileges to entries in an access control list (ACL) of an object.

account. Data in the Registry database that allows a principal to log in. An account is a registry object that relates to a principal.

ACF. Attribute configuration file.

address. An unambiguous name, label, or number that identifies the location of a particular entity or service. See *presentation address*.

Application Support Server. Refers to the server for z/OS DCE Application Support. The Application Support server allows a client program to access CICS or IMS.

architecture. (1) The organizational structure of a computer system, including the interrelationships among its hardware and software. (2) The logical structure and operating principles of a computer network. The operating principles of a network include those of services, functions, and protocols.

attribute. (1) RPC: An Interface Definition Language (IDL) or attribute configuration file (ACF) that conveys information about an interface, type, field, parameter, or operation. (2) DTS: A qualifier used with DTS commands. DTS has four attribute categories: characteristics, counters, identifiers, and status.
(3) XDS: Information of a particular type concerning an object and appearing in an entry that describes the object in the directory information base (DIB). It denotes the attribute's type and a sequence of one or more attribute values, each accompanied by an integer denoting the value's syntax.

attribute configuration file (ACF). RPC: An optional companion to an interface definition file that changes how the Interface Definition Language (IDL) compiler locally interprets the interface definition. See also *interface definition* and *Interface Definition Language*.

attribute type. (1) XDS: The component of an attribute that indicates the type of information given by that attribute. Because it is an object identifier, it is unique among other attribute types. (2) XOM: Any of various categories into which the client dynamically groups values on the basis of their semantics. It is an integer unique only within the package.

attribute value. XDS, XOM: A particular instance of the type of information indicated by an attribute type.

authentication. In computer security, a method used to verify the identity of a principal.

Authentication Service. One of three services provided by the Security Service: it verifies principals according to a specified authentication protocol. The other Security services are the Privilege Service and the Registry Service.

authorization. (1) The determination of a principal's permissions with respect to a protected object. (2) The approval of a permission sought by a principal with respect to a protected object.

B

binding. RPC: A relationship between a client and a server involved in a remote procedure call.

binding handle. RPC: A reference to a binding. See *binding information*.

binding information. RPC: Information about one or more potential bindings, including an RPC protocol sequence, a network address, an endpoint, at least one transfer syntax, and an RPC protocol version number. See *binding*. See also *endpoint*, *network address*, *RPC protocol*, *RPC protocol sequence*, and *transfer syntax*.

broadcast. A notification sent to all members within an arbitrary grouping such as nodes in a network or threads in a process. See also *signal*.

C

cache. (1) CDS: The information that a CDS clerk stores locally to optimize name lookups. The cache contains attribute values resulting from previous lookups, as well as information about other clearinghouses and namespaces. (2) Security: Contains the credentials of a principal after the DCE login. (3) GDS: See *DUA cache*.

CCITT. Consultative Committee on International Telegraphy and Telephone

CDS. Cell Directory Service.

CDS clerk. The software that provides an interface between client applications and CDS servers.

cell. The basic unit of operation in the distributed computing environment. A cell is a group of users, systems, and resources that are grouped around a common purpose and that share common DCE services.

Cell Directory Service (CDS). A DCE component. A distributed replicated database service that stores names and attributes of resources located in a cell. CDS manages a database of information about the resources in a group of machines called a DCE cell.

clerk. (1) DTS: A software component that synchronizes the clock for its client system by requesting time values from servers, calculating a new time from the values, and supplying the computed time to client applications. (2) CDS: A software component that receives CDS requests from a client application, ascertains an appropriate CDS server to process the requests, and returns the results of the requests to the client application.

client. A computer or process that accesses the data, services, or resources of another computer or process on the network. Contrast with *server*.

client context. RPC: The state within an RPC server generated by a set of remote procedures and maintained across a series of calls for a particular client. See *context handle*. See also *manager*.

client/server model. A form of computing where one system, the client, requests something, and another system, the server, responds.

clock. The combined hardware interrupt timer and software register that maintains the system time.

collapse. CDS: To remove the contents of a directory from the display (close it) using the CDS Browser. To collapse an open directory, double-click on its icon. Double-clicking on a closed directory expands it. Contrast with *expand*.

Consultative Committee on International Telegraphy and Telephone (CCITT). A United Nations Specialized Standards group whose membership includes common carriers concerned with devising and proposing recommendations for international telecommunications representing alphabets, graphics, control information, and other fundamental information interchange issues.

context handle. RPC: A reference to state (client context) maintained across remote procedure calls by a server on behalf of a client. See *client context*.

control access. CDS: An access right that grants users the ability to change the access control on a name and to perform other powerful management tasks, such as replicate a directory or move a clearinghouse.

copy. GDS, XDS: Either a copy of an entry stored in other DSAs through bilateral agreement or a locally and dynamically stored copy of an entry resulting from a request (a cache copy).

courier. DTS: A local server that requests a time value from a randomly selected global server. The time value returned is used for synchronization.

credentials. Security: A general term for privilege attribute data that has been certified by a trusted privilege certification authority.

cross-linking information. In order for z/OS DCE to provide RACF-DCE interoperability and single sign-on to DCE, DCE provides utilities (see **mvsexpt** and **mvsimpt**) to incorporate into RACF the information that associates a z/OS-RACF user ID with a DCE principal's identifying information and the DCE principal's UUID with the corresponding z/OS-RACF user ID. The information is placed in a RACF DCE segment and the RACF general resource class, DCEUUIDS. This is called **cross-linking information** and is what allows interoperability and single sign-on to work. See also *interoperability* and *single sign-on*.

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

D

daemon. (1) A long-lived process that runs unattended to perform continuous or periodic system-wide functions such as network control Some daemons are triggered automatically to perform their task; others operate periodically. An example is the **cron** daemon, which periodically performs the tasks listed in the **crontab** file. Many standard dictionaries accept the spelling *demon*. (2) A DCE server process.

DCE. Distributed Computing Environment.

directory. (1) A logical unit for storing entries under one name (the directory name) in a CDS namespace. Each physical instance of a directory is called a replica. (2) A collection of open systems that cooperates to hold a logical database of information about a set of objects in the real world.

directory ID. Directory identifier.

directory information tree (DIT). GDS: The directory information base (DIB) considered as a tree, whose vertices (other than the root) are the directory entries.

Directory Service. A DCE component. The Directory Service is a central repository for information about resources in a distributed system. See *Cell Directory Service* and *Global Directory Service*.

distinguished name (DN). GDS: One of the names of an object, formed from the sequence of RDNs of its object entry and each of its superior entries.

distinguished value. GDS: An entry's attribute value that has been designated to appear in the RDN of the entry.

distributed computing. A type of computing that allows computers with different hardware and software to be combined on a network, to function as a single computer, and to share the task of processing application programs.

Distributed Computing Environment (DCE). A comprehensive, integrated set of services that supports the development, use, and maintenance of distributed applications. DCE is independent of the operating system and network; it provides interoperability and portability across heterogeneous platforms.

Distributed Time Service (DTS). A DCE component. It provides a way to synchronize the times on different hosts in a distributed system.

DIT. Directory information tree.

DNS. Domain Name System.

Domain Name System (DNS). A hierarchical scheme for giving meaningful names to hosts in a TCP/IP network.

domain name. A unique network name that is associated with a network's unique address.

drift. DTS: The change in a clock's error rate over a specified period of time.

DUA cache. GDS: The part of the DUA that stores information to optimize name lookups. Each cache contains copies of recently accessed object entries as well as information about DSAs in the directory.

E

endpoint. RPC: An address of a specific server instance on a host.

endpoint map. RPC: A database local to a node where local RPC servers register binding information associated with their interface identifiers and object identifiers. The endpoint map is maintained by the endpoint map service of the DCE daemon.

endpoint map service. RPC: A service that maintains a system's endpoint map for local RPC servers. When an RPC client makes a remote procedure call using a partially bound binding handle, the endpoint map service looks up the endpoint of a compatible local server. See *endpoint map*.

entry. GDS, XDS: The part of the DIB that contains information relating to a single directory object. Each entry consists of directory attributes.

exception. (1) An abnormal condition such as an I/O error encountered in processing a data set or a file. (2) One of five types of errors that can occur during a floating-point exception. These are valid operation, overflow, underflow, division by zero, and inexact results. [OSF] (3) Contrast with *interrupt, signal*.

expand. CDS: To display the contents of (open) a directory using the CDS Browser. A directory that is closed can be expanded by double-clicking on its icon. Double-clicking on an expanded directory collapses it. Contrast with *collapse*.

F

foreign cell. A cell other than the one to which the local machine belongs. A foreign cell and its binding information are stored in either GDS or the Domain Name System (DNS). The act of contacting a foreign cell is called intercell. Contrast with *local cell*.

full name. CDS: The complete specification of a CDS name, including all parent directories in the path from the cell root to the entry being named.

fully bound binding handle. RPC: A server binding handle that contains a complete server address including an endpoint. Contrast with *partially bound binding handle*.

G

Global Directory Service (GDS). A DCE component. A distributed replicated directory service that provides a global namespace that connects the local DCE cells into one worldwide hierarchy. DCE users can look up a name outside a local cell with GDS.

global name. A name that is universally meaningful and usable from anywhere in the DCE naming environment. The prefix /... indicates that a name is global.

global server. DTS: A server that provides its clock value to courier servers on other cells, or to DTS entities that have failed to obtain the specified number of servers locally.

group. (1) RPC: A name service entry that corresponds to one or more RPC servers that offer common RPC interfaces, RPC objects, or both. A group contains the names of the server entries, other groups, or both that are members of the group. See *NSI group attribute*. (2) Security: Data that associates a named set of principals that can be granted common access rights. See *subject identifier*.

H

handle. RPC: An opaque reference to information. See *binding handle*, *context handle*, *interface handle*, *name service handle*, and *thread handle*.

heterogeneous. Pertaining to a collection of dissimilar host computers such as those from different manufacturers. Contrast with *homogeneous*.

home cell. Synonym for *local cell*.

homogeneous. Pertaining to a collection of similar host computers such as those of one model or one manufacturer. Contrast with *heterogeneous*.

host ID. Synonym for *network address*.

I

Information Management System (IMS). A database and data communication system capable of managing complex databases and networks in virtual storage.

interoperability. The capability to communicate, execute programs, or transfer data among various functional units in a way that requires the user to have little or no knowledge of the unique characteristics of those units.

instance. XOM: An object in the category represented by a class.

integrity. RPC: A protection level that may be specified in secure RPC communications to ensure that data transferred between two principals has not been changed in transit.

interface. RPC: A shared boundary between two or more functional units, defined by functional characteristics, signal characteristics, or other characteristics, as appropriate. The concept includes the specification of the connection of two devices having different functions. See *RPC interface*.

interface definition. RPC: A description of an RPC interface written in the DCE Interface Definition Language (IDL). See *RPC interface*.

Interface Definition Language (IDL). A high-level declarative language that provides syntax for interface definitions.

interface handle. RPC: A reference in code to an interface specification. See *binding handle* and *interface specification*.

interface specification. RPC: An opaque data structure that is generated by the DCE IDL compiler from an interface definition. It contains identifying and descriptive information about an RPC interface. See *interface definition*, *interface handle*, and *RPC interface*.

interface UUID. RPC: The Universal Unique Identifier (UUID) generated for an RPC interface definition using the UUID generator. See *interface definition* and *RPC interface*.

International Organization for Standardization (ISO). An international body composed of the national standards organizations of 89 countries. ISO issues standards on a vast number of goods and services including networking software.

Internet Protocol (IP). In TCP/IP, a protocol that routes data from its source to its destination in an Internet environment. IP provides the interface from the higher level host-to-host protocols to the local network protocols. Addressing at this level is usually from host to host.

ISO. International Organization for Standardization

K

Kerberos. The authentication protocol used to carry out DCE private key authentication. Kerberos was developed at the Massachusetts Institute of Technology.

key. A value used to encrypt and decrypt data.

L

LAN. Local area network.

local. (1) Pertaining to a device directly connected to a system without the use of a communication line. (2) Pertaining to devices that have a direct, physical connection. Contrast with *remote*.

local area network (LAN). A network in which communication is limited to a moderate-sized geographical area (1 to 10 km) such as a single office building, warehouse, or campus, and which does not generally extend across public rights-of-way. A local network depends on a communication medium capable of moderate to high data rate (greater than 1Mbps), and normally operates with a consistently low error rate.

local cell. The cell to which the local machine belongs. Synonymous with *home cell*. Contrast with *foreign cell*.

local server. DTS: A server that synchronizes with its peers and provides its clock value to other servers and clerks in the same network.

login facility. A Security Service facility that enables a principal to establish its identity.

M

manager. RPC: A set of remote procedures that implement the operations of an RPC interface and that can be dedicated to a given type of object. See also *object* and *RPC interface*.

master replica. CDS: The first instance of a specific directory in the namespace. After copies of the directory have been made, a different replica can be designated as the master, but only one master replica of a directory can exist at a time. CDS can create, update, and delete object entries and soft links in a master replica.

mvsexpt. One of two (the other is **mvsimpt**) utilities used to automate much of the administrator's work in creating the cross-linking information for DCE-RACF interoperability. The **mvsexpt** utility creates the cross-linking information in the RACF database from information in the DCE registry. See also *cross-linking information*, *interoperability*, and *single sign-on*.

mvsimpt. One of two (the other is **mvsexpt**) utilities used to automate much of the administrator's work in creating the cross-linking information for DCE-RACF interoperability. The **mvsimpt** utility creates DCE principals from information obtained from the RACF database. See also *cross-linking information*, *interoperability*, and *single sign-on*.

N

name. GDS, CDS: A construct that singles out a particular (directory) object from all other objects. A name must be unambiguous (denote only one object); however, it need not be unique (be the only name that unambiguously denotes the object).

name service. A central repository of named resources in a distributed system. In DCE, this is the same as Directory Service.

name service handle. RPC: An opaque reference to the context used by the series of next operations called during a specific name service interface (NSI) search or inquiry.

namespace. CDS: A complete set of CDS names that one or more CDS servers look up, manage, and share. These names can include directories, object entries, and soft links.

network. A collection of data processing products connected by communications lines for exchanging information between stations.

network address. An address that identifies a specific host on a network. Synonymous with *host ID*.

Network Data Representation (NDR). RPC: The transfer syntax defined by the Network Computing Architecture. See *transfer syntax*.

network protocol. A communications protocol from the Network Layer of the Open Systems Interconnection (OSI) network architecture, such as the Internet Protocol (IP).

Network Time Protocol (NTP). A clock synchronization protocol commonly used on an Internet.

node. (1) An endpoint of a link, or a junction common to two or more links in a network. Nodes can be preprocessors, controllers, or workstations, and they can vary in routing and other functional capabilities. (2) In network topology, the point at an end of a branch. It is usually a physical machine.

null time provider. The daemon that fetches the time from the hardware clock of the DCE host for DTS.

NSI group attribute. RPC: An RPC-defined attribute (NSI attribute) of a name service entry that stores the entry names of the members of an RPC group and identifies the entry as an RPC group. See *group*.

NTP. Network Time Protocol.

NULL. In the C language, a pointer that does not point to a data object.

O

object. (1) A data structure that implements some feature and has an associated set of operations. (2) RPC: For RPC applications, anything that an RPC server defines and identifies to its clients using an object Universal Unique Identifier (UUID). An RPC object is often a physical computing resource such as a database, directory, device, or processor. Alternatively, an RPC object can be an abstraction that is meaningful to an application, such as a service or the location of a server. See *object UUID.* (3) XDS: Anything in the world of telecommunications and information processing that can be named and for which the directory information base (DIB) contains information. (4) XOM: Any of the complex information objects created, examined, changed, or destroyed by means of the interface.

object UUID. RPC: The Universal Unique Identifier (UUID) that identifies a particular RPC object. A server specifies a distinct object UUID for each of its RPC objects. To access a particular RPC object, a client uses the object UUID to find the server that offers the object. See *object*.

OEM. Original equipment manufacturer.

Open Software Foundation (OSF). A nonprofit research and development organization set up to encourage the development of solutions that allow computers from different vendors to work together in a true open-system computing environment.

operation. (1) GDS: Processing performed within the directory to provide a service, such as a read operation. (2) RPC: The task performed by a routine or procedure that is requested by a remote procedure call.

organization. (1) The third field of a subject identifier. (2) Security: Data that associates a named set of users who can be granted common access rights that are usually associated with administrative policy.

original equipment manufacturer (OEM). A manufacturer of equipment that may be marketed by another manufacturer.

OSF. Open Software Foundation.

P

partially bound binding handle. RPC: A server binding handle that contains an incomplete server address lacking an endpoint. Contrast with *fully bound binding handle*.

password. A secret string of characters shared between a computer system and a user. The user must specify the character string to gain access to the system.

platform. The operating system environment in which a program runs.

port. (1) Part of an Internet Protocol (IP) address specifying an endpoint. (2) To make the programming changes necessary to allow a program that runs on one type of computer to run on another type of computer.

position (within a string). XOM: The ordinal position of one element of a string relative to another.

position (within an attribute). XOM: The ordinal position of one value relative to another.

presentation address. An unambiguous name that is used to identify a set of presentation service access points. Loosely, it is the network address of an open systems interconnection (OSI) service.

principal. Security: An entity that can communicate securely with another entity. In the DCE, principals are represented as entries in the Registry database and include users, servers, computers, and authentication surrogates.

privacy. RPC: A protection level that encrypts RPC argument values. in secure RPC communications.

privilege service. Security: One of three services provided by the Security Service; the Privilege Service certifies a principal's privileges. The other services are the Registry Service and the Authentication Service.

programming interface. The supported method through which customer programs request software services. The programming interface consists of a set of callable services provided with the product.

programming services. DCE services used mainly by programmers to develop distributed applications. Programming services include RPC and Threads.

proprietary. Pertaining to the holding of the exclusive legal rights in making, using, or marketing a product.

protocol. A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication.

protocol sequence. Synonym for *RPC protocol sequence*.

R

RACF. Resource Access Control Facility.

read-only replica. (1) CDS: A copy of a CDS directory in which applications cannot make changes. Although applications can look up information (read) from it, they cannot create, change, or delete entries in a read-only replica. Read-only replicas become consistent with other, changeable replicas of the same directory during skulks and routine propagation of updates. (2) Security: A replicated Registry server.

register. (1) RPC: To list an RPC interface with the RPC runtime. (2) To place server-addressing information into the local endpoint map. (3) To insert authorization and authentication information into binding information. See *endpoint map* and *RPC interface*.

Registry Service. Security: One of three services provided by the Security Service; the Registry Service manages information about principals, accounts, and security policies. The other services are the Privilege Service and the Authentication Service.

relative distinguished name (RDN). GDS, XDS: A set of Attribute Value Assertions (AVAs).

remote. Pertaining to a device, file or system that is accessed by your system through a communications line. Contrast with *local*.

remote procedure. RPC: An application procedure located in a separate address space from calling code. See *remote procedure call*.

remote procedure call. RPC: A client request to a service provider located anywhere in the network.

Remote Procedure Call (RPC). A DCE component. It allows requests from a client program to access a procedure located anywhere in the network.

replica. CDS: A directory in the CDS namespace. The first instance of a directory in the name space is the master replica. See *master replica* and *read-only replica*.

replication. The making of a shadow of a database to be used by another node. Replication can improve availability and load-sharing.

request. A command sent to a server over a connection.

resource. Items such as printers, plotters, data storage, or computer services. Each has a unique identifier associated with it for naming purposes.

Resource Access Control Facility (RACF). An IBM licensed program, that provides for access control by identifying and verifying the users to the system, authorizing access to protected resources, and logging the detected unauthorized access to protected resources.

ROM. Read-only memory.

RPC interface. A logical group of operations, data types, and constant declarations that serves as a network contract for a client to request a procedure in a server. See also *interface definition* and *operation*.

RPC protocol. An RPC-specific communications protocol that supports the semantics of the DCE RPC API and runs over either connectionless or connection-oriented communications protocols.

RPC protocol sequence. A valid combination of communications protocols represented by a character string. Each RPC protocol sequence typically includes three protocols: a network protocol, a transport protocol, and an RPC protocol that works with the network and transport protocols. See *network protocol*, *RPC protocol*, and *transfer protocol*. Synonymous with *protocol sequence*.

S

Security Service. A DCE component that provides trustworthy identification of users, secure communications, and controlled access to resources in a distributed system.

segment. One or more contiguous elements of a string.

server. (1) On a network, the computer that contains programs, data, or provides the facilities that other computers on the network can access. (2) The party that receives remote procedure calls. Contrast with *client*.

service. In network architecture, the capabilities that the layers closer to the physical media provide to the layers closer to the end user.

signal. Threads: To wake only one thread waiting on a condition variable. See *broadcast*.

sign-on. (1) A procedure to be followed at a terminal or workstation to establish a link to a computer. (2) To begin a session at a workstation. (3) Same as log on or log in.

simple name. CDS: One element in a CDS full name. Simple names are separated by slashes in the full name.

single sign-on. In z/OS DCE, single sign-on to DCE allows a z/OS user who has already been authenticated to an external security manager, such as RACF, to be logged in to DCE. DCE does this automatically when a DCE application is started, if the user is not already logged in to DCE.

specific. XOM: The attribute types that can appear in an instance of a given class, but not in an instance of its superclasses.

standard. A model that is established and widely used.

string. An ordered sequence of bits, octets, or characters, accompanied by the string's length.

subject identifier (SID). A string that identifies a user or set of users. Each SID consists of three fields in the form person.group.organization. In an account, each field must have a specific value; in an access control list (ACL) entry, one or more fields may use a wildcard.

subordinate. GDS, XDS: In the directory information tree (DIT), an entry whose distinguished name includes that of the other as a prefix.

synchronization. DTS: The process by which a Distributed Time Service entity requests clock values from other systems, computes a new time from the values, and adjusts its system clock to the new time.

sysplex. Systems complex. Multiple z/OS systems connected together to perform the processing for an installation.

system time. The time value maintained and used by the operating system.

T

thread. A single sequential flow of control within a process.

thread handle. RPC: A data item that enables threads to share a storage management environment.

Threads Service. A DCE component that provides portable facilities that support concurrent programming. The threads service includes operations to create and control multiple threads of execution in a single process and to synchronize access to global data within an application.

time provider (TP). DTS: A process that queries universal time coordinated (UTC) from a hardware device and provides it to the server.

time provider interface (TPI). An interface between the DTS server and external time provider process.

The DTS server uses the interface to communicate with the time provider and to obtain timestamps from an external time source.

time provider program. DTS: An application that functions as a time provider.

transaction. (1) A unit of processing consisting of one more application programs initiated by a single request, often from a terminal. (2) IMS/ESA: A message destined for an application program.

transfer syntax. RPC: A set of encoding rules used for transmitting data over a network and for converting application data to and from different local data representations. See also *Network Data Representation*.

Transmission Control Protocol (TCP). A communications protocol used in Internet and any other network following the U.S. Department of Defense standards for inter-network protocol. TCP provides a reliable host-to-host protocol in packet-switched communication networks and in an interconnected system of such networks. It assumes that the Internet Protocol is the underlying protocol. The protocol that provides a reliable, full-duplex, connection-oriented service for applications.

Transmission Control Protocol/Internet Protocol (TCP/IP). A set of non-proprietary communications protocols that support peer-to-peer connectivity functions for both local and wide area networks.

type. XOM: A category into which attribute values are placed on the basis of their purpose. See *attribute type*.

type UUID. RPC: The Universal Unique Identifier (UUID) that identifies a particular type of object and an associated manager. See also *manager* and *object*.

U

Universal Unique Identifier (UUID). RPC: An identifier that is immutable and unique across time and space. A UUID can uniquely identify an entity such as an object or an RPC interface. See *interface UUID*, *object UUID*, and *type UUID*.

user. A person who requires the services of a computing system.

UUID. Universal unique identifier

V

value. XOM: An arbitrary and complex information item that can be viewed as a characteristic or property of an object. See *attribute value*.

vector. RPC: An array of references to other structures.

vendor. Supplier of software products.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

VTAM. Virtual Telecommunications Access Method.

W

workstation. A device that enables users to transmit information to or receive information from a computer, for example, a display station or printer.

X

X.500. The CCITT/ISO standard for the open systems interconnection (OSI) application-layer directory. It allows users to register, store, search, and retrieve information about any objects or resources in a network or distributed system.

Bibliography

This bibliography is a list of publications for z/OS DCE and other products. The complete title, order number, and a brief description is given for each publication.

z/OS DCE Publications

This section lists and provides a brief description of each publication in the z/OS DCE library.

Overview

- *z/OS DCE Introduction*, GC24-5911

  This book introduces z/OS DCE. Whether you are a system manager, technical planner, z/OS system programmer, or application programmer, it will help you understand DCE and evaluate the uses and benefits of including z/OS DCE as part of your information processing environment.

Planning

- *z/OS DCE Planning*, GC24-5913

  This book helps you plan for the organization and installation of z/OS DCE. It discusses the benefits of distributed computing in general and describes how to develop plans for a distributed system in a z/OS environment.

Administration

- *z/OS DCE Configuring and Getting Started*, SC24-5910

  This book helps system and network administrators configure z/OS DCE.

- *z/OS DCE Administration Guide*, SC24-5904

  This book helps system and network administrators understand z/OS DCE and tells how to administer it from the batch, TSO, and shell environments.

- *z/OS DCE Command Reference*, SC24-5909

  This book provides reference information for the commands that system and network administrators use to work with z/OS DCE.

- *z/OS DCE User's Guide*, SC24-5914

  This book describes how to use z/OS DCE to work with your user account, use the directory service, work with namespaces, and change access to objects that you own.

Application Development

- *z/OS DCE Application Development Guide: Introduction and Style*, SC24-5907

  This book assists you in designing, writing, compiling, linking, and running distributed applications in z/OS DCE.

- *z/OS DCE Application Development Guide: Core Components*, SC24-5905

  This book assists programmers in developing applications using application facilities, threads, remote procedure calls, distributed time service, and security service.

- *z/OS DCE Application Development Guide: Directory Services*, SC24-5906

  This book describes the z/OS DCE directory service and assists programmers in developing applications for the cell directory service and the global directory service.

- *z/OS DCE Application Development Reference*, SC24-5908

  This book explains the DCE Application Program Interfaces (APIs) that you can use to write distributed applications on z/OS DCE.

Reference

- *z/OS DCE Messages and Codes*, SC24-5912

  This book provides detailed explanations and recovery actions for the messages, status codes, and exception codes issued by z/OS DCE.

z/OS SecureWay® Security Server Publications

This section lists and provides a brief description of books in the z/OS SecureWay Security Server library that may be needed for z/OS SecureWay Security Server DCE and for RACF® interoperability.

- *z/OS SecureWay Security Server DCE Overview*, GC24-5921

  This book describes the z/OS SecureWay Security Server DCE and provides z/OS SecureWay Security Server DCE information about the z/OS DCE library.

- *z/OS SecureWay Security Server LDAP Client Programming*, SC24-5924

  This book describes the Lightweight Directory Access Protocol (LDAP) client APIs that you can use to write distributed applications on z/OS DCE and gives you information on how to develop LDAP applications.

- *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683.

  This book explains RACF concepts and describes how to plan for and implement RACF.

- *z/OS SecureWay Security Server LDAP Server Administration and Use*, SC24-5923

  This book describes how to install, configure, and run the LDAP server. It is intended for administrators who will maintain the server and database.

- *z/OS SecureWay Security Server Firewall Technologies*, SC24-5922

  This book provides the configuration, commands, messages, examples and problem determination for the z/OS Firewall Technologies. It is intended for network or system security administrators who install, administer and use the z/OS Firewall Technologies.

Tool Control Language Publication

- *Tcl and the Tk Toolkit*, John K. Osterhout, (c)1994, Addison—Wesley Publishing Company.

  This non-IBM book on the Tool Control Language is useful for application developers, DCECP script writers, and end users.

IBM C/C++ Language Publication

- *z/OS C/C++ Programming Guide*, SC09-4765

  This book describes how to develop applications in the C/C++ language in z/OS.

z/OS DCE Application Support Publications

This section lists and provides a brief description of each publication in the z/OS DCE Application Support library.

- *z/OS DCE Application Support Configuration and Administration Guide*, SC24-5903

  This book helps system and network administrators understand and administer Application Support.

- *z/OS DCE Application Support Programming Guide*, SC24-5902

  This book provides information on using Application Support to develop applications that can access CICS® and IMS™ transactions.

Encina Publications

- *z/OS Encina Toolkit Executive Guide and Reference*, SC24-5919

  This book discusses writing Encina applications for z/OS.

- *z/OS Encina Transactional RPC Support for IMS*, SC24-5920

  This book is to help software designers and programmers extend their IMS transaction applications to participate in a distributed, transactional client/server application.

Index

Readers' Comments

z/OS
DCE
Planning

Publication No. GC24-5913-00

You may use this form to report errors, to suggest improvements, or to express your opinion on the appearance, organization, or completeness of this book.

Date: ________________________________

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Note

Report system problems to your IBM representative or the IBM branch office serving you.
U.S. customers can order publications by calling the IBM Software Manufacturing Solutions at **1-800-879-2755**.

In addition to using this postage-paid form, you may send your comments by:

FAX	1-607-752-2327	Internet	pubrcf@vnet.ibm.com
IBMLink	GDLVME(PUBRCF)		

Would you like a reply? ___**YES** ___ **NO** If yes, please tell us the type of response you prefer.

___ Electronic address: ________________________________

___ FAX number: ________________________________

___ Mail: (Please fill in your name and address below.)

Name

Address

Company or Organization

Phone No.

Readers' Comments
GC24-5913-00

Cut or Fold
Along Line

IBM®

Fold and Tape   **Please do not staple**   Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

Department G60
International Business Machines Corporation
Information Development
1701 North Street
ENDICOTT NY 13760-5553

Fold and Tape   **Please do not staple**   Fold and Tape

Cut or Fold
Along Line

Program Number: 5694-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

GC24-5913-00

Spine information:

IBM z/OS DCE Planning