1996 CTG Achievement Forum
Abstraction Paper

Proposed Paper Title: "DCE: A Standard to Watch"

Topic (Technical): The topic of this paper is the suite of technologies from OSF (Open Software Foundation) known as DCE (Distributive Computing Environment). This paper will provide an understanding of DCE, the suite of technologies which comprise DCE, and the added-value to CTG.

Abstraction:

DCE's main goal is to provide a cohesive open systems environment allowing for the seamless development of a distributive system. This transparent interoperabilty is built upon current operating systems, and can be integrated without disturbing existing applications. DCE consists of an integrated suite of technologies that reduce the complexity involved with creating, using, and maintaining applications in a distributive environment.

The objective of this paper is to educate CTG employees. Although this paper is a technical paper, it will take into account the professional diversity of the audiences. Technical terms will be communicated to non-technical people. From the information provided in this article, the audience will be better equipped to more effectively represent CTG in the dynamic market place.

This author has been fortunate enough to work on multiple projects involving DCE. He has participated in roles that include system architect, designer, and developer. This author understands the value of DCE first hand, and is excited with the opportunity to share the knowledge with his fellow CTGers. His personal goal is to present the topic in such a way that it excited the highly technical as well as the novice.

Name & Business Unit: Michael S. Pallos
Senior Consultant, Network Management
3820 Cason Road
Plant City, FL 33566
(813) 754-8391
VMX: 3291

Outline

DCE: A Standard to Watch

Michael S. Pallos
Senior Consultant, Network Management

I. Introduction

AS CTG continues to excel in the computer technologies industry, it is imperative that we continue learning and developing new skills. Continuing education is more important now more than ever before since technology is changing so rapidly. Within the world of client/server technologies, a suite of services know as Distributive Computing Environment (DCE) from the Open Software Foundation (OSF) is emerging as the industry standard for open computing.

Currently, DCE is one of the most viable options available in the current market place to aid in the development of enterprise network systems incorporating the client/server model. DCE provides a layer of transparency required to minimize the complexity of the distributive environment for the application developer and end user. OSF's DCE is the most comprehensive approach in developing client/server systems offering vast interoperablity in a heterogeneous environment.

II. History of DCE

The history of DCE begins with the formation of the Open Software Foundation. In 1988 many computer vendors, including; IBM, DEC, and Hewlett-Packard, joined forces and created OSF. OSF's main goal was to create a flavor of UNIX that would be more widely accepted than a proprietary version of UNIX developed by AT&T and SUN. The result was the UNIX operating system OSF/1. To facilitate the building of distributed applications, OSF's next challenge was to build a layer product which resided on top of OSF/1 and other UNIX systems, and would be used to build distributed applications. OSF sent out a Request For Technology (RFT) and received responses from over fifty vendors, (of which only 29 were OSF members). The selected technologies were mostly derived from the DECorum proposal jointly submitted by HP/Apollo, IBM, DEC, Microsoft, Locus, and Transarc.

With the selection of DECorum, OSF had positioned itself as an alternative to SUN's Network Computing Architecture (NAC). This decision was accelerated by IBM's announcements of the forthcoming interoperabilty between Systems Application Architecture (SAA) and Advanced Interactive eXecutive (AIX). This established the foundation for the integration of DCE, SAA and AIX.

III. Goals of DCE

DCE's main goal is to provide a cohesive open systems environment allowing for the seamless development of a distributive system. This transparent interoperabilty is built upon current operating systems, and can be integrated without disturbing existing applications. DCE consists of an integrated suite of technologies that reduce the complexity involved with creating, using, and maintaining applications in a distributive environment.

The distributive system in which DCE runs can be a heterogeneous system consisting of computers from multiple vendors, each with its own operating system (OS). DCE is layered on top of the operating system, allowing DCE to hide OS differences and convert data types when necessary. Because this is transparent to the application developer, the programmer can focus on the application being developed instead of the complexities of the architecture. The developers can work on the same system when developing from multiple computer environments.

DCE technologies will be incorporated by IBM (into MVS, AIX, DOS/Windows, and OS/2), DEC (into VMS, Ultrix, OSF/1, and ACE), HP (into HP/UX, Domain, and OSF/1), Gradient (into MS-DOS), Cray (into Unicos), Siemens (into SINIX), and Tandem (into Guardian and Integrity). DCE will also interoperate (at the RPC level) with Atlas, the competing open standard fro distributed computing from the now defunct UNIX International. (Orfali 149-150)

IV. DCE Components

DCE consists of seven major components. The key technology services comprising DCE include: Directory Service, Diskless Support Service, Distributed Time *synchronization* Service, Distributed File Service, Remote Procedure Calls, Security Service, and Threads Service.

The DCE Components

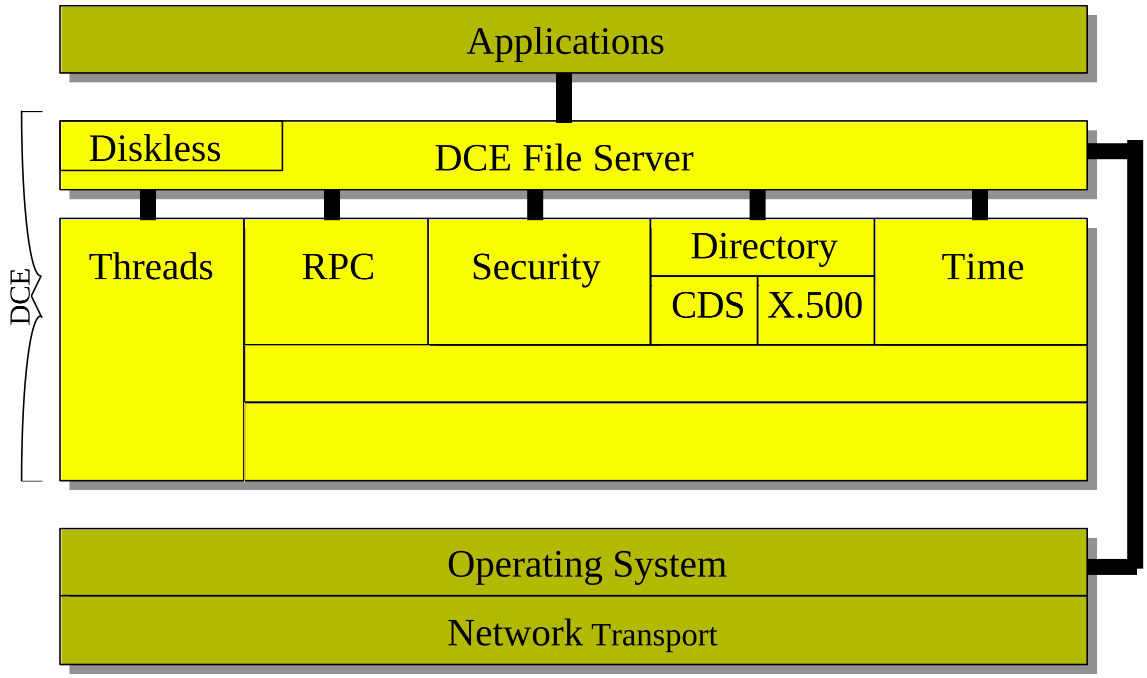

Figure 1.0 - The DCE Components

A. Directory Service

One of the challenges of a Distributed Computing Environment is the ability to identify and locate all of the available resources, (i.e., servers, clients, print queues, applications, files, disks, faxes, modems and other resources) without knowing their physical location. DCE Directory Server (sometimes referred to as Name Service) does this by dividing up the universe into domains called *cells*. A cell is an administrative unit which consists of clients, servers and resources. A cell usually contains a group of resources responsible for a specific task. Cells are defined by the customer (system administrator). At a minimum, a cell will consist of a directory server (name server) and a security server.

The Directory Service operates much like the telephone directories yellow pages. Given a persons name, the directory provides the phone number. When the Directory Service is given a resource unique name, the network address along with other information about the resource is returned. Currently, the main use of Directory Server is providing connectivity among DCE disturbed application clients with distributed remote servers.

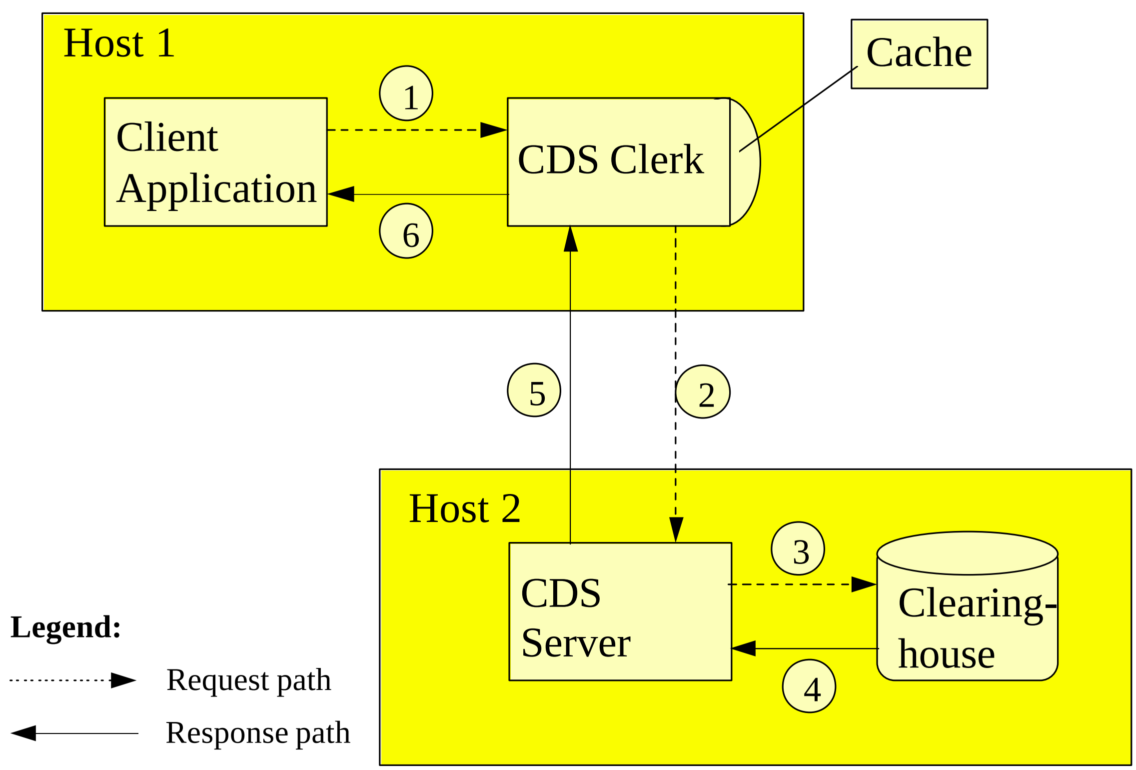

Figure 2.0 - A Simple CDS lookup (Rosenberry 81)

B. Diskless Support Service

In the event that a DCE client includes a diskless workstation, DCE helps diskless workstations connect to the DFS (Directory File Server). During process idle time, the diskless machine will need to store idle memory, i.e., swapping. DCE Diskless Support allows a diskless workstation the ability to swap its memory to a remote machine that contains a disk. Therefore, the diskless workstation connects to a DFS and the virtual disk is available for memory swapping.

C. Distributed Time Service

In distributed networks time synchronization is of the utmost importance. DCE's Disturbed Time Service provides a mechanism for maintain time synchronization across all WAN and LAN system clocks. The environment must contain at least three Time Servers with at least one of the Time Servers being connected to an External Timing Provider, e.g., a clock.

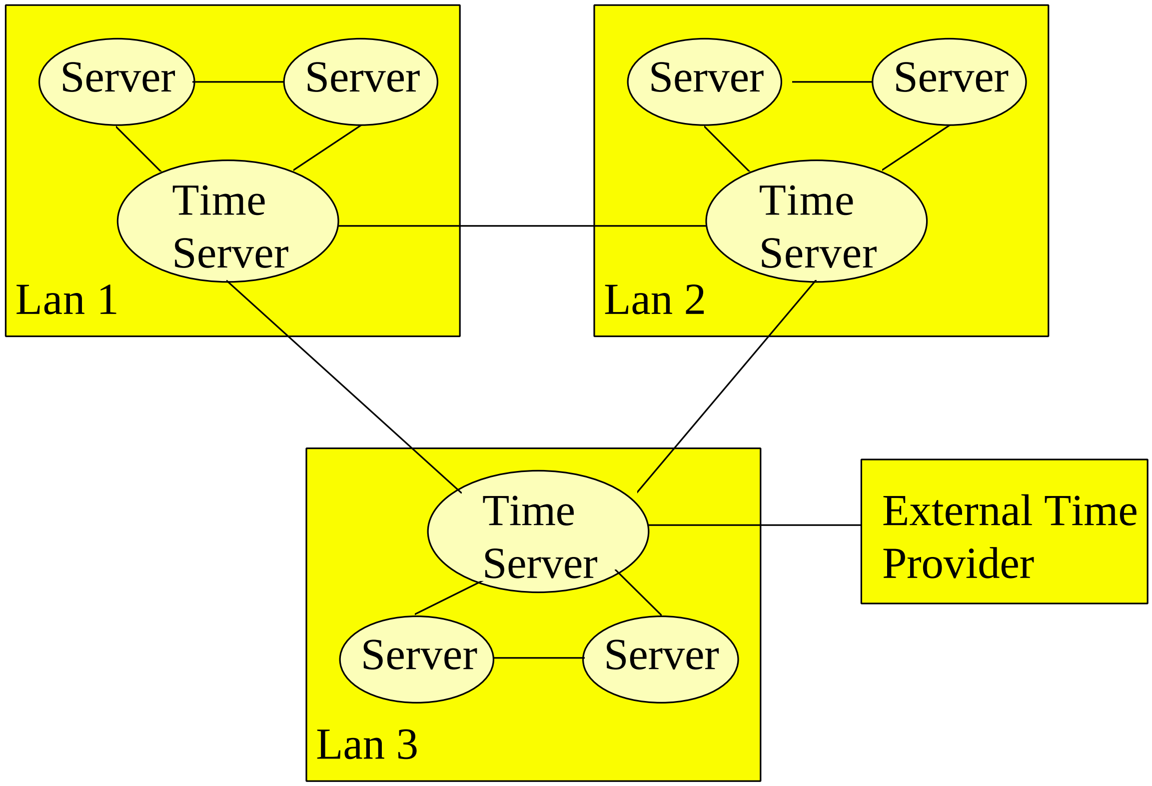

Figure 3.0 - Time Synchronization Service

D. Distributed File Service

The DCE Distributed File Server (DFS) provides file location transparency and high availability in a uniform name space. Files and directories can be invisibly replicated on multiple servers for high availability. With the incorporation of DCE Security, unauthorized users can be omitted from selected files, while authorized users can gain access. A cache-consistency protocol allows a file to be changed in cache. This change is propagated through the universe identifying all locations utilizing the file. Each file is given a unique identifier providing a simple image file system that can be distributed across the network. Because DFS fully utilized other DCE services, DFS administration can be administered from any node (maintaining security considerations).

E. Remote Procedure Calls

The foundation of DCE is the utilization of Remote Procedure Calls (RPC). RPCs provide a method for developing distributed applications, both client and server, in a self contained powerful and simple method. For the creation of RPC, an Interface Definition Language (IDL) and a compiler that facilitated the creation of the RPC is provided. The RPC author develops an IDL interface file. The IDL compiler reads in the interface file and generates a client and server stub. The stubs are in the C programming language and are object code. DCE requires the use of IDL, and fully supports the C language. One may use other languages, but the risk is higher. (For you *leading-edgers* idl++ is available for C++). IDL registers the client stub and the server stub with the DCE RPC runtime libraries. The client and server stubs are linked in with the application at compile time.

RPC are extremely powerful, and can control access to process when incorporating DCE Security Services. Also, servers can concurrently service RPC by incorporated threads. Utilizing DCE Directory Service with Security Server an RPC an authenticate each RPC and dynamically locate the requested server at runtime.

The one weakness in DCE's RPC is it's current inability to manage transactions. With the incorporation of a Transaction Processing (TP) Monitor, a transaction can be managed from its point of origin. From a client, across multiple servers, and back to the client. When the transaction ends, all parties involved must agree that the transaction either failed or succeeded. TP monitors are also required for a Two-Phase Commit. The two-phase commit protocol is used to synchronize updates on different machines so that either all fail or all succeed. Some of the major TP monitors available include; Encina RPC, OSI TP, Top End, Tuxedo, and X/Open.

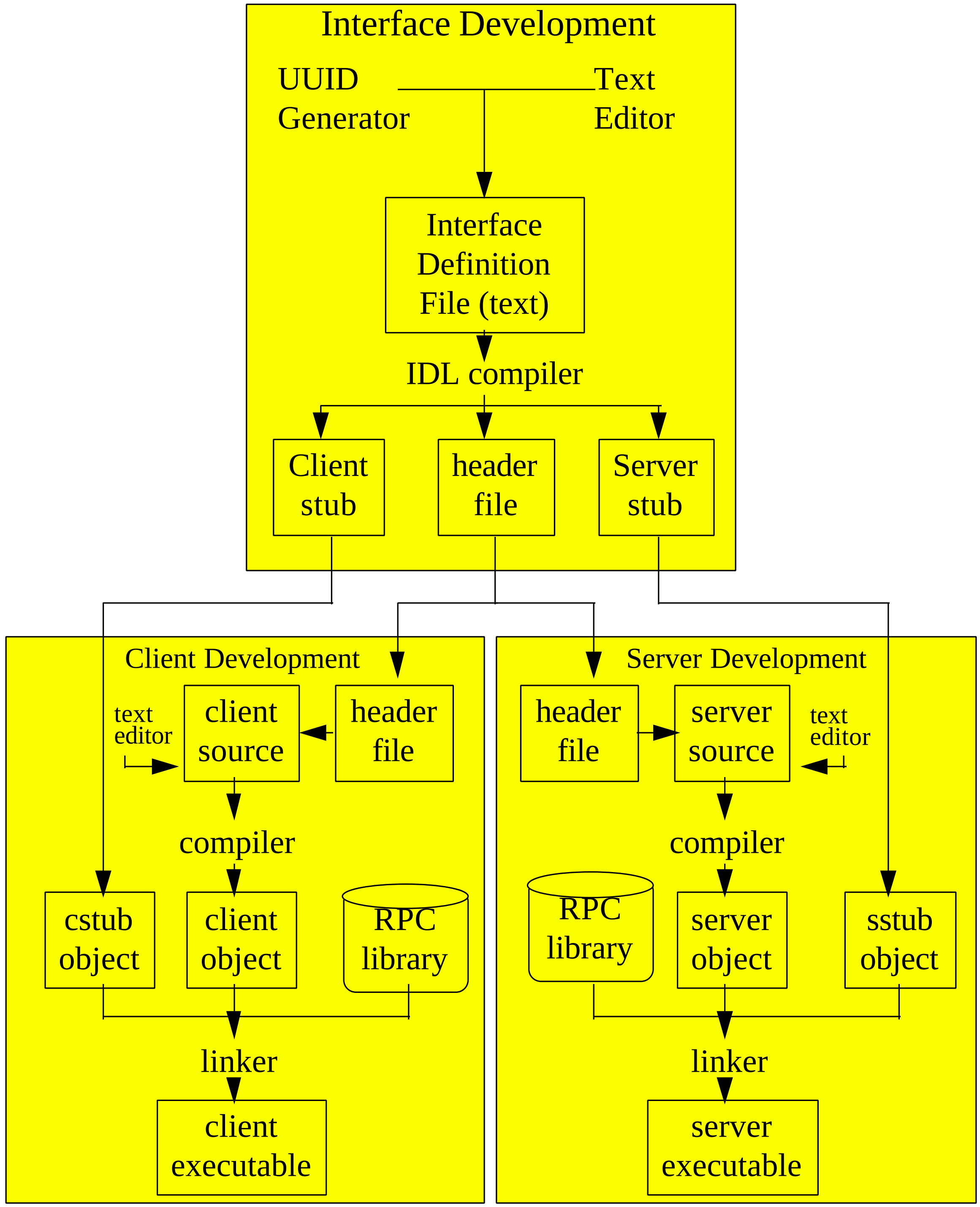

Figure 5.0 - Application development with RPC, both client and server.

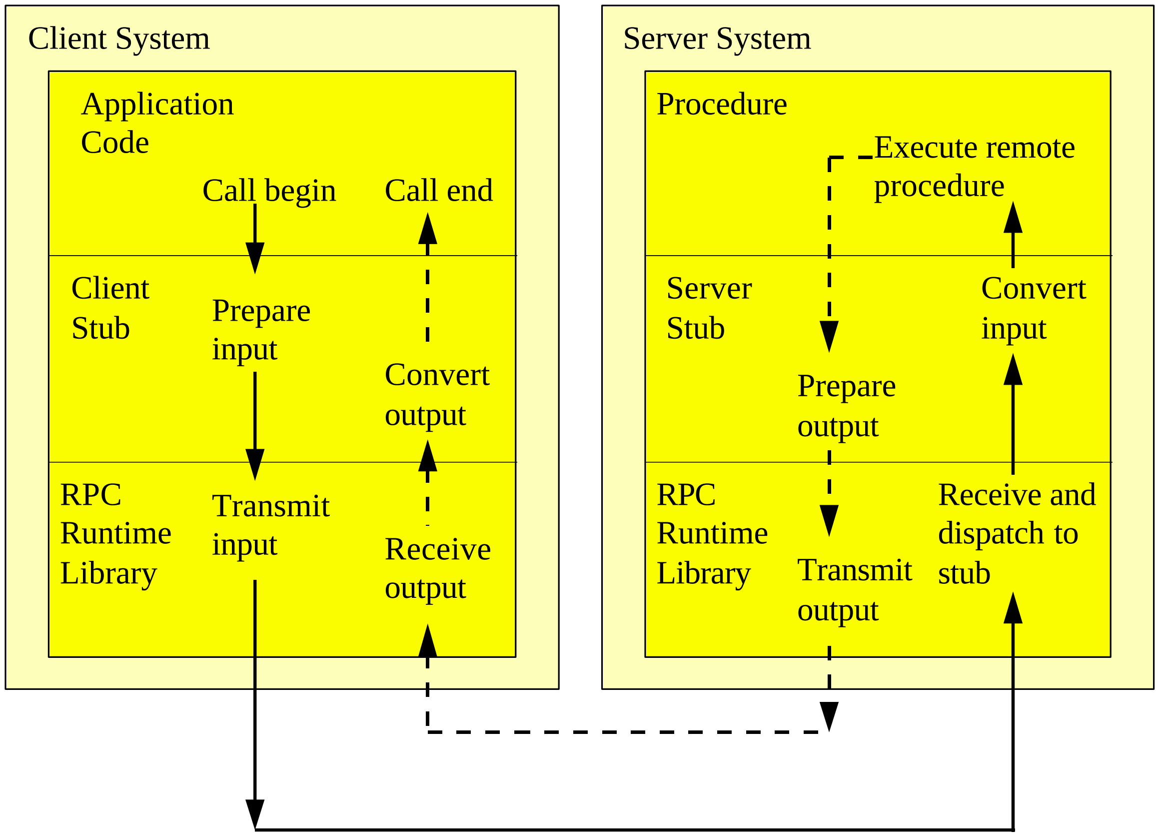

Figure 5.1 - Client/Server execution of an application with an RPC

F. Security Service

The DCE Security Service is based on MIT's Kerberos authentication system. The Security Service architecture is based on total mistrust and utilizes one central security database. The service verifies the principles authorization and authentication prior to allowing access to the requested resource. Security granularity can be taken to the most minuet detail. The level of security implementation is left up to the architecture and the high level design. Obviously a banking network may incorporate a more granular implementation than a public records system.

Kerberos is the three-headed mythological monster responsible for guarding the gates of Hades. The Security Service contains three components residing on the same box; Authentication Server, Security Database, and Privilege Server. Security validation is accomplished through secure communications capability provided by an RPC and the ticketing mechanism. All DCE machines must run a security agent. Each DCE cell must have a Security Server.

Some of the terminology used with security include:

Authentication	- validate the principle (login and password).
Authorization	- after the principle has been Authenticated does it have permission to access the requested resource?
Principle	- user, computer or server.
Units of Trust	- synonymous with Principle.

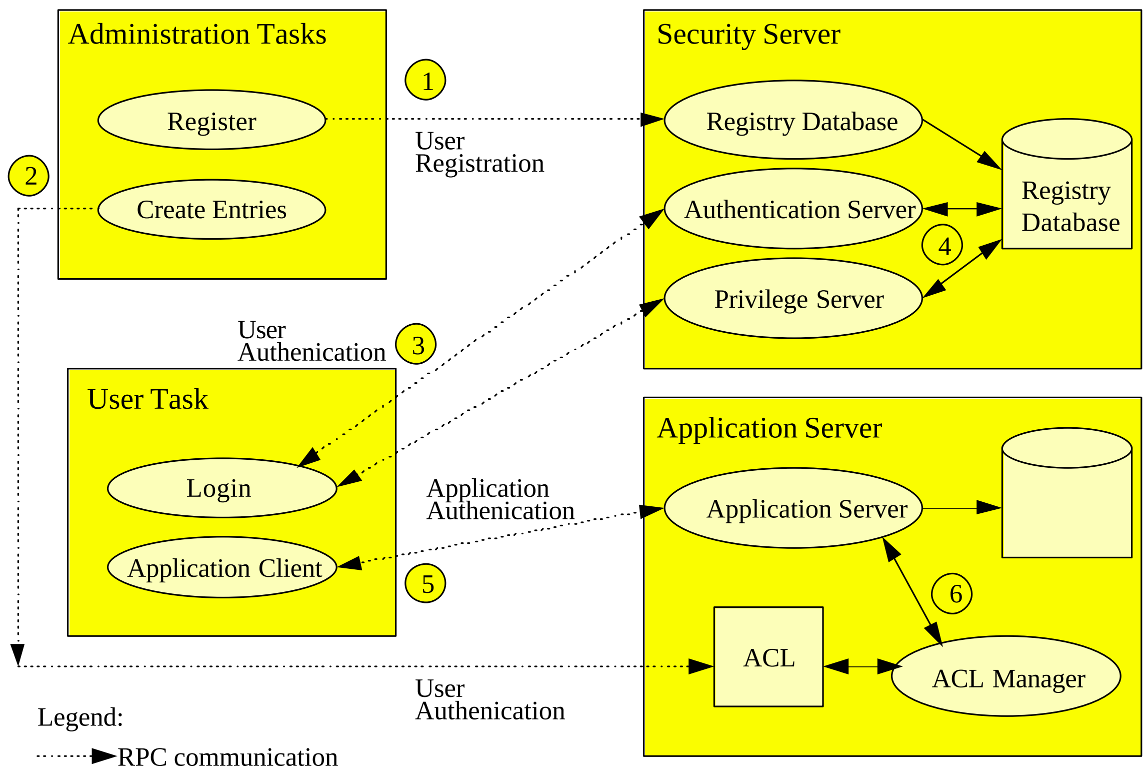

Figure 6.0 - General tasks in a security environment

With DCE Security, the level of authorization and authentication granularity is often left up the application architect or the application developer. As Figure 6.0 identifies, six steps have to be taken to perform a client to server request. Although value is added from a security perspective, it is not without cost to the processing time. i.e., the tighter the application is locked down, the more behind the scenes security processing and hand shaking taking place, the slower the application.

G. Threads

Threads afford the developer the luxury of supporting granular levels of multitasking from within an application. DCE Threads support multiprocessor environments using shared memory. This popular method of parallelism improves the applications performance, and reduces the time frame required for sequential entrance, processing, and exiting of a client request.

From a UNIX perspective, starting multiple threads is similar to forking a process (forking: a parent process spawns a child process). Although forking can be useful, in volume forking is undesirable. Forks spawn a separate process each time, with a large amount of forks an application can bring the system down to it's knees. Unlike forks, threads of a process share the same address space, static and external data, file handles, pipes, and anything else know to the whole process. Therefore, the benefits gained include; less memory, less time required for startup and context switching, and usually easier implementation of parallel-programming algorithms.

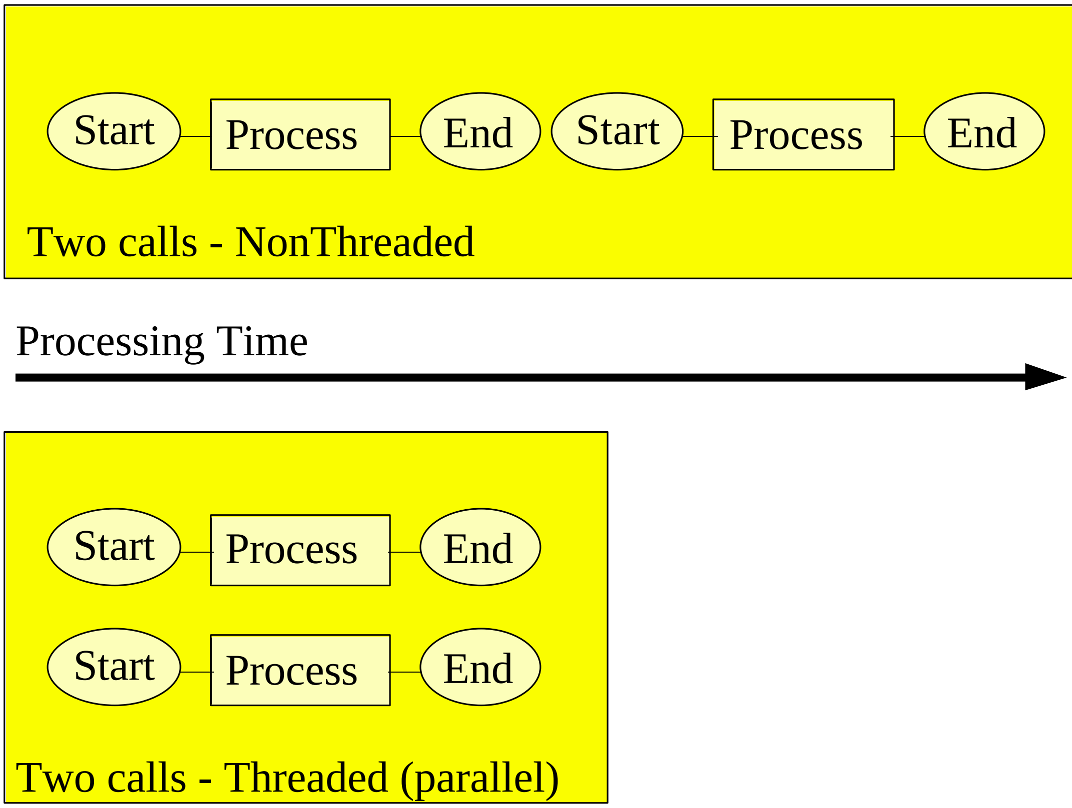

Figure 7.0 - Threaded verses NonThreaded Processing

V. Conclusion

Distributed Computing Environment (DCE) is the best available solution for developing and integrating multivendor clients, servers, and legacy systems in a heterogeneous client/server environments, and is positioned to become the industry standard. DCE's strength lies within its integration technology that allows a client process to interoperate with a server process when potentially both processes reside on different operating system from different vendors. Best of all, the middleware components are transparent to the application developer.

VI. Frequently Asked Questions

Q: With Sun Microsystems owning such a large share of the client/server market place, will DCE support Sun?

A: Yes, DCE has been ported to Solaris (Sun's OS) and is currently available.

Q: How does one purchase DCE.

A: DCE can not be purchased from OSF directly. One must purchase DCE from OSF licensees, who package and sell products.

Q: We have many PC applications on our desktops, such as word-processors and spreadsheets. How will DCE impact our current environment?

A: DCE allows users and applications programs to use the operating system exactly as they had done before the installation of DCE.

Q: When is the next release of DCE, and what's new.

A: DCE version 1.2 is due in December 1995. This release is scheduled to include linkage to Novell NetWare and Sun Microsystems Computer Company's Open Network Computing (ONC) architecture.

Q: Who is successfully using DCE today?

A: OSF's list of users currently conducting DCE tests includes Boeing, 3M, Citibank, and ITT Hartford.

Q: What might inhibit the adoption of DCE? What should I, Joe User, be aware of before developing a mission critical application with DCE?

A: Currently, there is not an adequate amount of tool kits and packaged applications available to assist with DCE development.

Q: What might inhibit the adoption of DCE? What should I, Joe User, be aware of before developing a mission critical application with DCE?

A: The following points are currently relevant:

- There is not an adequate amount of tool kits and packaged applications available to assist with DCE development.

- One weakness in DCE is it's current inability to manage transactions. Fortunately, this responsibility is handled with the incorporation of a Transaction Processing (TP) monitor. With a TP Monitor a transaction can be manage from its point of origin. From a client, across multiple servers, and back to the client. When the transaction ends, all parties involved must agree that the transaction either failed or succeeded. TP monitors are also required for a Two-Phase Commit. The two-phase commit protocol is used to synchronize updates on different machines so that either all fail or all succeed. Some of the major TP monitors available include; Encina RPC, OSI TP, Top End, Tuxedo, and X/Open.

- Full incorporation of DCE Security will slow application processing down considerably. With security, the level of authorization and authentication granularity is often left up the application architect or the application developer. As Figure 6.0 identifies, six steps have to be taken to perform a client to server request. Although value is added from a security perspective, it is not without cost to the processing time. i.e., the tighter the application is locked down, the more behind the scenes security processing and hand shaking taking place, the slower the application.

VII. Recommended Reading & References

Berson, Alex. Client/Server Architecture, McGraw-Hill, New York, 1992.

Boar, Bernerd. Implementing Client/Server Computing, AT&T, USA, 1993.

Kernighan, Brian W., & Dennis Ritchie. The C Programming Language, Printice-Hall, USA, 1988.

Orfali, Robert. Essential Client/Server Survival Guide, Van Nostrand Reinhold, USA, 1994

RosenBerry, Ward. Understanding DCE, O'Reilly and Associates, Sebastopol CA, 1994.

Shirly, John. Guide to Writing DCE Applications, O'Reilly and Associates, Sebastopol CA, 1992.

Smith, Pat. Client/Server Computing, SAMS Publisher, Indiana, 1992.

Spewak, Steven. Enterprise Architecture Planning, QED Publishing, Wesll, MA 1993.

Stroustrup, Bjarne. The Annotated C++ Reference Manual, Addison-Wesley Publishing Company, New York, 1990.

Tanenbaum, Andrew. Distributed Operating Systems, Prenstice/Hall, Englewood-Cliff, NJ, 1995.

Watterson, Karen. Client/Server Technologies for Managers, Addison-Wesley Publishers, Reading, MA, 1995.