Open Software Foundation
Request For Comments: 95.0
August 1996

Michael Guidry (Phillips Petroleum)
Mark Michael (Hughes Electronics)
William Estrem (3M)

DCE/NEXT: REQUIREMENTS SUMMARY

1. INTRODUCTION

1.1. So Far, So Good

The Open Software Foundation's Distributed Computing Environment (OSF/DCE) has made significant progress towards meeting the goals of providing a deployable and affordable distributed computing infrastructure that support a rich set of distributed application models supporting a highly scalable, heterogenous environment. It is currently well suited to provide support for intra-enterprise needs in areas such as:

(a) Location-independent naming.

(b) Secure, authenticated, authorized computing resource access.

(c) Assured, platform-independent data delivery.

(d) Time synchronization across large numbers of computer platforms.

(e) Single signon to DCE applications existing within a DCE cell.

(f) File access using globally understood names.

It is implemented by many suppliers on multiple operating system platforms. The delivery of reference implementation source code and a published Application Environment Specification (AES) has made consistent interoperability between platforms achievable. As each successive version of reference source code has been delivered to licensees, completion time of the port has declined.

So far, so good. However, there is work that remains to be completed in order for DCE to achieve the goal of the premier global computing environment. As DCE has developed, the world has not stood still. Likewise, as the users of DCE have gained experience, their requirements have evolved.

The current OSF/DCE (Revision 1.2) still meets many of these requirements, but there remains a need to provide additional features and functions. The purpose of this document is to provide a basis for discussion of the end user requirements, the characteristics of next generation DCE, and establishing consensus by the affected communities.

1.2. What Should be Next (DCE/Next)

Recently, one of the authors of this RFC found a way to describe where DCE should go next, in just four words, borrowing from a recent campaign slogan:

> ‘‘It's the Internet, stupid!’’

There is rapidly increasing interest using the Internet for commerce. There are two kinds of Internet commerce: consumers' browsing and purchasing goods from Internet marketeers' Web sites, and inter-enterprise trusted communications between commerce partners — what Bob Metcalfe (co-inventor of Ethernet and co-founder of 3Com) has termed an ‘‘extranet’’. DCE has very nearly all of the components needed to provide for such communications (security, trust, etc.) DCE/Next must complete the picture, in order for DCE to continue to be relevant.

Until recently, enabling DCE for Internet usage and growing DCE into the premier object-oriented platform would have been parts of two separate discussions. With the recent activity surrounding the Java programming language and environment, the relationship between the two has become clearer: object support within intranets, the Internet and extranets will require all of the services now built into DCE in order for such application of object technology to be successful.

The proponents of DCE have two choices: either watch the rest of the world re-create the services currently found in DCE, and watch DCE become irrelevant to the emerging planet-wide computing infrastructure; or, rapidly grow

DCE into the premier supplier of these services, enabling DCE licensees to inexpensively deploy these next-generation applications using DCE/Next.

2. TARGET

This document is intended for general review and approval by the membership of the OSF DCE SIG and Open Group's DCE Task Group. It is provided as a means for the membership to state in writing what we think OSF/DCE should include in future releases that is not already in the currently available releases of OSF/DCE. Other RFCs that are germane are referenced, but this document is closest in spirit to RFC 63.* — the difference being that this document specifies requirements input to the DCE/Next project (and beyond), whereas RFC 63.2 details committed plans of the DCE 1.2 project. (As of this writing, there are no committed plans for DCE/Next, so this document is timely.)

3. GOALS AND NON-GOALS

3.1. Goals

(a) Provide a document that reflects the requirements for DCE/Next based as accurately as possible on what the membership would like to see included in DCE/Next.

(b) Guide future development of DCE/Next in an open process based on one of the following, in order of descending preference based on pragmatic considerations:

   (i) The Open Group development process.

   (ii) A release train (e.g., 1.3.1, 1.3.2, etc.) as products from a PST.

   (iii) A coordinated series of mini-PSTs.

   (iv) A coordinated series of ATOs.

   (v) An RFT, but with a special emphasis on rapid completion of the requirements definition phase (typically where RFTs become painfully slow).

   In all cases, any projects commenced to perform the work described herein will always have its source code bundled into the license DCE source tree.

(c) Leverage and extend the DCE in an controlled and evolutionary manner, while providing continued support for the base of functionality found in DCE 1.2. Any change to DCE that is not completely forward-compatible must have a clearly understood forward migration path, and must allow for a mixed DCE 1.2-DCE/Next environment during migration.

(d) Improve DCE's market acceptance by including features making it better capable of supporting its use both within enterprise intranets and beyond, through the Internet, to interoperate with DCE and other services in other, Internet-connected enterprise networks.

(e) Actively support the introduction of distributed object technology into DCE-based computing environments.

(f) Preserve DCE's current strengths in secure, reliable data transport, while adding support for interoperability with public-key authentication and encryption mechanisms being adopted for other Internet-based protocols and services.

(g) Further improve DCE's support for three-tier environments and load-balanced server access.

(h) Further enhance DCE's interoperability with popular proprietary networked services such as ONC and NetWare, without compromising DCE's existing strengths.

(i) Make DCE even more affordable to purchase, deploy/install, configure, manage and utilize.

3.2. Non-Goals

(a) Break from the past so completely that existing DCE applications are broken.

4. ORGANIZATION OF THIS DOCUMENT

The remainder of this document is divided into the following sections:

(a) Environment.

(b) Attributes.

(c) Threads.

(d) Communications.

(e) Security.

(f) Directory.

(g) Administration.

(h) Distributed File System.

(i) Tools.

(j) Applications.

(k) Branding and Metrics.

A few clean-up sections then finish the document.

5. ENVIRONMENT

The challenge that DCE/Next must meet will include:

(a) **Mobile computing.**

The next generation of Internet protocols (under the umbrella of IP version 6 and Mobile IP) are being designed to allow for computer systems to interoperate with other Internet-accessible computers. DCE/Next must aggressively support these; by the year 2000, it is likely that computer users will need support from satellite-communications-enabled laptops, for example.

(b) **Movable computing.**

Movable computing is not the same as mobile computing. In movable computing, an office worker's computer may be relocated with him or her when he or she moves to another office. If that computer's IP address changes due to the move, the DCE administrator(s) must take manual steps to re-enable that computer for many of the more advance DCE-related functions. IP version 6 has been designed with many options for automatic address assignment, including but not limited to Dynamic Host Configuration Protocol (DHCP). DCE/Next must account for this environment, preferably by integration with services supporting movable computing, or alternatively a superior set of substitute services.

(c) **Palmtop computing.**

Personal digital assistant (PDA) computers have been through several generations since their introduction. Today, increasingly specialized ''wearable computers'' are now coming to market. DCE/Next will need to accomodate their special needs and constraints. For example, it may be possible to use an active-mode smart card to automatically authenticate the user presenting it; the device controlled by a DCE application could just as easily be a security door lock as a desktop computer.

(d) **The Internet.**

Beyond Web browsers enabling virtual shopping malls, there are an increasing number of commercial software applications being adapted to use Web browsers for their user interface. However, many of these applications still have client-server security services requirements. The other services included in the existing DCE are also seen as valuable once they are better explained. However, DCE support by the firewall vendor community is critical; without it, extranet builders will look elsewhere for solutions. [See also section 9(a), DCE relations with SET (Visa, Masterdard).]

(e) **Planet-wide applications deployment.**

The original Decorum project's goal was to provide ''a file system for the planet''. That file system is now the DFS component of DCE. It is time for DCE to realize that goal; in fact, one of the primary goals of DCE 1.2 is to enable large-scale DCE deployment. The technology included in DCE 1.2 is part of the solution; getting the cost of DCE licensing, installation, deployment, use and support down to a fraction of what it is today must be a continuing goal. Most desktop users today do not yet have access to DCE at a level that their enterprise IT departments consider to be as affordable as providing that same user with a TCP/IP capability and a Web browser. If DCE is to avoid irrelevance, this situation must improve quickly.

(f) **Object-oriented programming support and interoperability.**

The members of the Object Management Group (OMG), Microsoft and SunSoft have each raised the ''object-consciousness'' of the global computing community. OMG has spent the last five years developing the Common Object Requestor-Broker Architecture (CORBA). Microsoft has been developing what it calls (as of this writing) the Distributed Common Object Model (DCOM). SunSoft, for its part, has led the effort to develop the Java language and environments to execute Internet-delivered Java runtime modules. DCE/Next should provide the premier method for end users to run applications based on these object models, or risk a situation where an enterprise IT organization must choose between spending limited resources on a DCE infrastructure or one that supports their selected object model, with DCE often ending up on the losing end of such a selection.

6. ATTRIBUTES

Some key attributes for DCE/Next that will facilitate its growth are:

(a) **Pluggable components.**

Each of the major components of DCE can be thought of as having subsystems within them. For example, the DCE Security Service as of DCE 1.1 and earlier only supported the usage of Kerberos protocol for pre-authentication and authentication. In DCE 1.2, some support of public-key authentication has been added, as part of moving towards support of peer-to-peer DCE services (anticipated as needed for DCE to successfully support object services). Having the ability to substitute one service subsystem for another is a requirement for many intranet users in order for them to use DCE.

(b) **Object model.**

DCE should be the premier infrastructure upon which to implement CORBA, DCOM or other object services; alternatively, DCE should provide its own object services.

(c) **Address renumbering.**

As part of the Internet Engineering Task Force's (IETF) work to improve the Internet Protocol (IP) suite, working groups within the IETF have been working on dynamic assignment of IP addresses. The existing version (IPv4) of IP has been augmented by the development of Dynamic Host Configuration Protocol (DHCP); the next version of IP (IPv6) includes a DHCPv6, plus several other methods for automatically generating globally unique IPv6 host addresses. A growing number of Internet-based computers and appliances will be programmed to use dynamic addressing, and thus will have their addresses renumbered frequently and automatically. DCE must be augmented to support this.

(d) **Freedom from export restrictions.**

The existing versions of DCE are actually two reference implementations: one that contains the source code for implementing packet-level privacy (which uses the Data Encryption Standard [DES] algorithm), and another version that does not implement packet-level privacy. For DCE to be successfully used for global commerce, it must have a packet-level privacy implementation (weaker than DES) capable of being acceptable for export by the U.S. government, yet still meets business requirements of organizations (a non-trivial feat!).

(e) **Component choices.**

With the advent of pluggable components, it will become possible to have alternative implementations of some DCE protocols and services. The most frequently requested service needing alternative implementations is the DCE Security Service, both its authentication and packet-privacy (encryption) features. Another popular request has been to allow for the substitution of a local directory service other than CDS in DCE. The reference implementation of DCE should include a robust selection of pluggable components for those services where the capability has been implemented; the provision of component choices should not be left solely up to the implementing vendors — this will lead to a splintering of the interoperability of DCE, which is one of its chief strengths.

(f) **CORBA interoperability.**

The Object Management Group (OMG) has been developing a series of standards as part of the Common Object Requestor-Broker Architecture (CORBA). Many of the services to be included in CORBA have yet to be finalized (network security is an example); it would be wise to make DCE more capable of supporting CORBA implementations to sit atop DCE's services. Another approach would be to include wire-protocol interoperability between DCE services and CORBA services, where appropriate.

(g) **MS-RPC interoperability.**

Arguably, the most widely deployed implementation of the DCE RPC (with some known shortcomings) is the Microsoft RPC embedded within Windows NT and Windows 95. Taking further steps to improve DCE's support for the existing MS-RPC implementations would greatly facilitate more widespread use of DCE-based applications and services.

(h) **Single point of management.**

Originally, DCE was to be managed by DME. That hasn't happened. The existing DCE management tools are still somewhat disjoint. A single, unified, command-and-GUI-based utility set is a popular request. Even with the creation of the `dcecp` program, opportunities exist for further improvement, especially in the area of integrating DFS management into `dcecp`.

(i) **Convergence with like standards.**

Much work has been done by other standards-making organizations in areas such as directory, security and authentication services and protocols. DCE should be further improved to include support for these. [*Editor's note:* This needs to be made more specific, or dropped. We do refer to activities elsewhere in the document that would achieve the goals set forth in this section.]

7. THREADS

Four issues are of concern for DCE/Next support of IEEE 1003.1x, known as POSIX threads:

(a) **DCE to POSIX threads.**

OSF began work on OSF/DCE using a draft of the IEEE POSIX definition for threads that has since been superseded by a ratified version. Clearly, OSF/DCE must now migrate from the existing threads library (''DCE threads'') to the new thread model (''POSIX threads'').

This migration is non-trivial, but holds many benefits. An increasing number of operating systems are now providing POSIX threads natively in their kernels. DCE is bound to run much more efficiently in memory and CPU usage by making this transition. The Open Group reference implementation must be upgraded to use POSIX threads.

(b) **Application migration.**

Migrating the DCE source code alone without considering what might happen to existing DCE applications could break existing DCE-thread-safe applications. Care must be taken to assure that existing applications are provided a well-documented migration path that is as simple and painless as possible.

(c) **Native threads use.**

If an operating system platform provides POSIX threads natively (typically, within the operating system kernel), DCE should be capable of utilizing it readily, transparent to DCE applications. This should result in a very substantial performance improvement over the existing DCE threads implementation, which typically runs in operating systems' user-mode rather than within the kernel.

(d) **DCE threads support.**

DCE threads support should continue to be available via compile-time options. Within DCE threads support, it should be possible to select translation of DCE threads to POSIX threads, or continuance of using user-mode DCE threads. This will allow the DCE community to migrate over time to POSIX threads, with a minimum of disruption due to unexpected application behavior caused by migrating DCE to POSIX threads, while allowing applications that can do so to take advantage of the expected performance improvements gained by using native POSIX threads.

8. COMMUNICATIONS

The services and protocols that DCE uses are being upgraded to support the growth of the Internet. Likewise, services and protocols existing above DCE require new services from DCE as well as extensions of the existing services.

(a) **DNS, IP renumbering and IPv6 support.**

As part of support of movable and mobile computing, there is a need for reliable reassignment of a computer's IP address. The Internet Engineering Task Force has working groups on Mobile IP (as a functional area) and within the IP version 6 effort.

How can DCE (better) handle renumbering? One contributor's reading of IETF RFC 1900 says that they consider mobility of IP address to be the same thing. Work should begin very soon to determine the impact of renumbering on DCE, if any. The same contributor raised an important question about the latency period between when the IP infrastructure renumbers a host and when DCE could propagate the change into all existing service definitions containing the old IP address. Originally, one of the authors had advocated simply using DNS fully-qualified domain names as a valid host identification within a protocol tower definition and within string bindings; however, the same contributor makes an excellent point that there is no reason to believe that one directory service will track reality more closely than another. Should DCE use DNS names instead of IP addresses? This is a significant change to the current philosophy and should be studied carefully. Again it is not clear that RFC 1900 is referring to a case like DCE. They talk about IP addresses in human readable configuration files. Is adding a level of indirection between CDS and the network a good idea? It would appear not, but this question is one that deserves a carefully-thought-out answer, possibly including some testing of the alternatives in software.

For example, one possible solution is for the DCE Directory Service to have the services it provides extended to include Domain Name Services itself, including the newest extensions for dynamic DNS updates and authenticating DNS update requests. The framework for such an augmentation of DCE is clearly already present, and this idea may be worth further discussion.

Does DCE require changes to support IPv6, with its increased address length, vastly larger address space, authentication and encryption options, quality-of-service support and dynamic readdressing of end systems? This is an important issue that needs to be researched carefully. Early adoptors of DCE are likely to be early adoptors of IPv6. Since any needed changes to DCE may take several (3) years to reach the field, this should get immediate attention.

(b) **Protocol support.**

What are DCE RPC towers and string bindings for and what should they contain? Should they be different representations of the same thing or should they be used for different things and contain different information? DCE is very fuzzy in this area. For example, OODCE takes the view that a binding is a reference to an object instance, but clearly this is not true in the case of replication. On the other hand, if you are using context, this is key. A subsidiary question is to what extent should towers and string bindings be opaque and to what extent should examining their contents be sanctioned?

[*Editor's note:* Would adding support for SNA/APPN protocols be interesting to some of the alternative mainframe vendors, such as Amdahl?]

(c) **Single-threaded RPC.**

Recent discussions on expanding the applicability of DCE have included proposed ATOs for the development of a DCE implementation so compact in size that it could be used in embedded systems. Many of these implementations would work better with a single-threaded programming model. The initial work to include a single-thread client in the DCE reference source code has been done in DCE 1.2; that work should be expanded to support embedded DCE, and other systems capable only of running a single-threaded DCE implementation.

(d) **Transactional RPC.**

One contributor has stated that a great deal of the RPCs that his company is coding end up looking very much like transactions. The work that has been done to date on defining extensions to the DCE RPC to support transactions in the X/Open TxRPC work should be completed and a reference implementation included in the core DCE code.

[*Editor's note:* Should we ensure any transactional RPC is conformant with the X/Open DTP model CRM? Also, should we adapt the RPC to support OSI Transaction Types?]

(e) **Support for messaging in DCE, including asynchronous RPC.**

When OSF completed work on the original DCE RFT, provision of an asynchronous RPC capability was left strictly up to the underlying platform supporting the use of DCE threads. Since that time, it has become clear that a simpler programming model is needed. DCE 1.2 provides a single-thread client for use on simpler operating systems; however, it still assumes a synchronous RPC model, in which the client application waits for the return of each RPC call.

There is still a need for direct support of messaging within DCE. A number of vendor-proprietary messaging products have been enhanced to use the DCE Security and Directory services; if the underlying RPC service is enhanced to support an asynchronous RPC (where the DCE call returns to the calling program instead of waiting for the RPC to complete execution first), this will greatly increase the attractiveness of DCE as a messaging services, and as the messaging layer for object services.

(f) **Support for message queueing in DCE.**

To complete the messaging metaphor within DCE/Next will require the inclusion of a reliable delivery services for messaging data. This is typically implemented via a message queueing service robust enough to be restartable without losing a message, and assuring delivery of those messages already acknowledged to the calling application, barring a disaster which prevents the recall for completion of messages stored in the queue.

(g) **Integration of DCE with IPC services.**

Allowing same-host DCE applications to use the fastest means available to them for exchanging the equivalent of RPC packet payloads via local host services such as interprocess communications (IPC). It would be wise to allow for such high-performance data exchange without the need for the programmers to account for it in their software implementation; DCE should handle it for them at a minimal penalty.

(h) **Multicast and broadcast RPC.**

There is a class of application services not yet addressed within the DCE RPC: transmission of RPCs to multiple recipient systems via a single RPC call. DCE/Next should include an investigation as to the practicality of including both multicast (transmission to a group of multiple systems that is a subset of all reachable hosts) and broadcast (transmission to all reachable hosts) RPCs.

(i) **Cross-cell services.**

For DCE to be viewed more practically as an extranet solution, better facilities for delegated administration in a cross-cell environment must be provided. In addition, the ability to treat DCE principals and other objects as transparent members of multiple cells (perhaps via a cross-cell aliasing facility) is needed.

9. SECURITY

As a general reference to this section (and to other security-related issues in the present document), see the Security SIG's excellent set of requirements (OSF-RFC 8.*).

(a) **Web commerce security integration.**

Full support and/or interoperability with Secure Electronic Transactions (SET), Secure Sockets (SSL), etc., having to do with Internet commerce.

(b) **Encryption.**

The export version of DCE source code ships with ‘‘crypto with a hole’’ — that is, the *calls* to DES are in the code, so the only thing missing is the actual DES code itself (which is easy to obtain from many sources, e.g., as printed source code in books). This makes it very easy for foreign vendors to implement full DCE, and OSF has received no creditable complaints from foreign vendors that it is too difficult for them to put DES back into DCE. Nevertheless, it is sometimes claimed (mostly by end-users) that the principle obstacle to deployment of DCE outside the U.S. is the restriction on the export of source code implementing the encryption features. Until a mechanism is provided to overcome this regulatory problem, one of the greatest features of DCE (its security) is perceived (by end-users) as being unavailable. This effectively limits DCE's deployment to within the U.S., until DCE includes support for U.S.-exportable encryption mechanisms (weaker than DES). (The exportability of *binaries*, both from the U.S. and from other countries, is another matter, beyond the scope of this document.)

(c) **Public key support.**

The use of Public Key technology in the integration of One-Time-Passwords (OTP) and Kerberos (or DCE) authentication. A good amount of public key technology is being added in OSF/DCE 1.2 release for authentication. Future releases of DCE should complete any work in this area which failed to make it into the release.

(d) **Configurable authentication mechanisms.**

A mechanism has to be included into DCE Security that allows for the site-specific implementation and configuration of the authentication mechanisms. To give a specific example, one customer is using personal chipcards for user authentication, and host-specific chipcards for machine authentication. Both chipcards are accessed through GSS-API. In order to integrate with DCE (to give DCE a chance for large scale deployment), we need an interface to plug our chipcards into the DCE security framework, plus a configuration switch that lets the installation choose at runtime which authentication mechanism to use. PAM (Pluggable Authentication Module) covers the idea, but an exact spec is requested on how to integrate custom built authentication mechanisms. This has to be field-installable.

10. DIRECTORY

(a) **Support usage of DCE by more directory clients.**

As one contributor puts it: ‘‘We've got far too many directories at the present time. I've done some looking and it appears that we could persuade WinNT to look at a CDE directory — haven't tried it but the book on DCE & NT from O'Reilly gives a hack for it. Still a product from someone, best if it were part of DCE, to integrate with MS's directory would be nice. A completely open directory service, where we could use our choice of directories that had been engineered to fit in with DCE would be very helpful. If the XFN PST achieves that, we'll be very happy.’’

[*Editor's note:* NSI support, included within DCE 1.1 and onward, may provide some of this support.]

(b) **LDAP Support.**

Recently, marketplace momentum has picked up considerably behind the use of the Lightweight Directory Access Protocol for directory browsing (typically by an end user via an application such as a Web browser) or program interface (such as an electronic mail application's need for a shared address book.)

OSF/DCE's directory service should support LDAP (RFC 1730) in two forms: straight IETF RFC-compliant LDAP with optional RFC 1510-compliant Kerberos authentication, and the LDAP protocol tunneled within the DCE RPC, or otherwise integrated with DCE security services. In the former case, an existing application which supports LDAP may use the DCE directory; in the latter, the same applications can continue to operate, but should utilize a DCE client when present.

Interoperability demonstrations should include popular electronic mail products whose clients are based upon Internet standards and use LDAP as their shared address book. This may require enhancement of the base-level DCE directory services schema to accomodate such interoperability; if so, it is part of the requirement for such a project to be considered successful.

(c) **Attribute-based searching.**

The capability to perform attribute-based searchs through the directory would be very helpful.

(d) **BLOBS.**

Providing user defined data types (including BLOBs) for storage in CDS would help. One interesting usage of this would be to include a rasterized photograph of the principal as an attribute of a defined principal.

(e) **More.**

Improved intercell features (fixing ERA's, adding global groups/users) and simplifying master/slave, CDS initial/replica change over in case of failure were also requested.

11. ADMINISTRATION

(a) **Single point of management.**

In section 12(b) the interoperability with Netware is discussed. It is a disappointment that the single-point-of-management has been deleted from the original proposal (RFC 63.0). We have heard from a potential major Australian end-user that this single fact is likely to affect their decision to adopt the DCE technology. As deployment is a stated goal of DCE 1.2 it is necessary for both sections 12(b), (c) to include a single-point-of-management tool.

Absence of management tools is a perceived major failing of DCE and consequently a barrier to deployment. Extending the improvements made in DCE 1.1 should be a goal for 1.2. Especially helpful would be to integrate DFS administrative functions with `dcecp`.

[*Editor's note:* Include support for policy/role based administration?]

(b) **Fully functional cell manager accessible via secure web browser.**

Originally, DCE cells were to be managed via OSF/DME. That no longer appears to be viable. Therefore, better management of DCE resources in its absence is required. The existing Cell Manager in DCE should be enhanced so that it is fully functional for administering all DCE objects, and can be run as a Web server using DCE-Web technology as developed by the OSF RI. This assures secure access, yet provides for a management user interface that is readily understood by any Net-literate computer user.

(c) **Performance and instrumentation.**

RFC's 32 and 33 provide some of the requirements for DCE performance measurement and instrumentation. Implementing the recommendations described in those RFC's should be a priority; many DCE implementations are growing large enough that performance troubleshooting, and ideally reliable forecasting of cell performance, is becoming an increasingly important issue. Having access to a benchmark would be nice, or a method described for benchmark development and execution, with a means for sharing and exchanging benchmark methods and data.

(d) **Hierarchical cells / transitive trust.**

Get hierarchical cells (transitive trust) working if it is technically feasible. If not, put a fork in it and let people know so that they can do their cell architectures accordingly. Also, support one-way trust between cells.

[*Editor's note:* Get SNMP, Events, and Managed Objects into the reference implementation?]

12. DISTRIBUTED FILE SYSTEM

(a) **Administration.**

We have found that while AFS and DFS both offer superior scalability to other distributed filesystems, there are still some administrative problem areas. The most notable to us is in the accounting, tracking and management of fileset (or volumes). This becomes extremely apparent in larger cells (those over 500GB).

Ideally, what every admin probably wants is a robust HSM (hierarchical storage management) system like those found on an IBM MVS Mainframe or like Pittburgh's Multi-resident AFS system. However, we are not naive enough to believe that this is can be easily delivered for DFS by OSF, given the number of vendors that are producing DFS products (and striving to differentiate themselves from one another).

With that being the case, it seems as though the best course would be to standardize an API and database for storing the type of information needed to effectively track and manage DFS filesets. We honestly believe that if such a thing existed, then it would be easier to convince HSM vendors (like Legato, Epoch, IBM ADSM, etc.) to produce systems, which could cleanly support DFS. In the worst case, it would at least provide a facility for the admins to produce reusable tools to automate this accounting process.

What we want therefore, is some sort of database where we could record information apropos to fileset management. This would include things like: who a fileset was created for, why it was created (comments), what its backup and replication policy are, what the lifetime of the data is (when should it be taken off-line), etc.

If you look at this, you will quickly realize that it is very similar to the EAs (Extended Attributes) associated with the Registry and `dced` namespaces. Since DCE 1.1 already has APIs and code internally available to support EAs, it seems that attaching EAs to FileSet Objects would be a good idea. (This is what we would like to see! Actually, attaching EAs to File and Directory objects would also be useful, but that's another topic.)

If for some reason EAs for DFS Filesets are prohibitive, another approach might be to extend the Ubik database, used in the Backup System. Given the state of DFS APIs (and our own experience with the ever-changing AFS APIs) versus those found in the DCE Core, we personally prefer the EA approach.

Regardless, we feel that some sort of database for fileset accounting, tracking and management is badly needed.

(b) **NetWare 4 support in DFAM.**

The recent introduction of Distributed File Access Method in DCE 1.2 has provided a considerable improvement in the ability to deploy DFS into a NetWare 3.∗-based environment. However, many NetWare users are now migrating to NetWare 4. Some OSF vendor members are incorporating support into their DCE products that allow for interoperability or even integration between NetWare 4 environments and DCE; it would be appropriate for DFA to be updated to support NetWare 4 also. Novell has introduced the Novell Directory Service as a replacement for the NetWare 3 bindery service, and a new generation of client software; determining the cost and duration of a project to further enhance DFA to provide support for these new products would be welcomed, given the relatively limited life-span contemplated for the existing NetWare 3-oriented DFA subsystem in DFS.

(c) **NFS 3 support in NFS gateway.**

Recently, several vendors have begun to include NFS version 3 in their versions of the Unix operating system. Determining the cost and duration of a project to further enhance the NFS Gateway portion of DFS to provide support for NFS 3-based clients would be welcomed, given the potential for existing NFS users to migrate to NFS 3.

[*Editor's note:* Eliminate the gateway approach to NFS/DFS connectivity?]

13. TOOLS

(a) **Development environment.**

Better tools for development: caching library, replication library as just some examples.

(b) **Application programming interface (API).**

An API set for general use by applications for logging or error reporting. This would allow applications, especially services, to present a message or general log to a general monitor or reporting service.

(c) **Object-oriented.**

We would like to see more extensive C++ support in DCE releases following 1.2.2; in particular, given the C++ support in 1.2.1 IDL, there should be a concerted effort to create a comprehensive C++ class library to simplify DCE programming.

Of course, we are promoting the adoption of HP OODCE into OSF DCE. Generally, we would like to see C++ interfaces for any new feature added to DCE. As Rich Salz has pointed out, XFN already has defined C++ interfaces; DCE's XFN support should include these interfaces. OSF's goal should be to make DCE the premier choice for distributed object-oriented programming.

The significance of user-to-user authentication is not evident while the DCE programming model remains heavily based on the server side of the client/server model. A more object-oriented system model could see the fruits of this proposal. An obvious comment though on user-to-user authentication is that it should be public-key based.

14. APPLICATIONS

(a) **Rebundling of DME print services into DCE.**

A question that inevitably comes up when explaining DFS to a new group of system administrators is, ‘‘what kind of print services does DCE support?’’ Today, the answer is, ‘‘none, without adding on stuff’’.

The source code for OSF/DME Print Services is readily available. It should readily interoperate with other Palladium-based print services (most next-generation print services are). It should be rebundled into OSF/DCE, with a project to perform the rebundling and whatever integration and updating is required. The license fees, especially for desktop clients, must not be impacted by this project, or its purpose will be voided.

(b) **Internet applications.**

The requirement for DCE to include a several common network applications such as TELNET and FTP was voted into the top ten requirements in OSF-RFC 8.1. The text of these requirements does not state that only the kerberized versions of these applications will be included. We were expecting that the FTP and TELNET applications released with DCE would be DCE-developed versions that take full advantage of the additional security services that DCE provides beyond those offered by Kerberos. The advantage of this approach is that network applications can use the richness of DCE authorisation, with PACs, groups, ACLs and UUID based identification. Kerberos on the other hand only offers named-based authentication.

(c) **Inclusion of DCE-Web in main source code library for DCE.**

As with many OSF Research Institute and OSF Advanced Technology Operation projects, the DCE-Web project's source code has not been integrated into the main DCE source code library. Using a model derived from that originally adopted for DFS (pre-1.1 DCE), the DCE-Web source code should be made an integrated part of DCE/Next, available upon payment of additional license fees, but with the source code kept up to date and synchronized with the rest of the DCE/Next source code.

(d) **Electronic mail (RFC 8.1).**

Secure email was the eleventh highest requirement of OSF-RFC 8.1 out of 104 requirements. For deployment reasons, it would be sensible to include a secure email application or the developement tools to build one. Some features of electronic mail (such as guaranteed delivery, connectivity, interoperability, availability and reliability, storage and retrieval, archival, survivability, directory services integration, scalable, security services integration, nonrepudiation of recipient, access control, entity-to-entity authentication, integrity, management services, accounting/billing, fault management, security management, systems planning, performance analysis, ease of use, timely deliver) may require infrastructure features.

Provision of a utility to support access to DCE encryption via MIME types would allow seamless integration with existing user applications.

[*Editor's note:* Perhaps it might be a good idea to provide a reference implementation based upon a popular (or, alternatively, a public-domain) MIME mailer in which the MIME encrypted and signed body parts have DCE-based options supported.]

(e) **Mobile computing.**

DCE has some serious inhibitors in this area, which need to be addressed. The whole notion of dynamic reconfiguration is foreign to DCE and state of practice in this environment.

(f) **Palmtop computing.**

DCE is getting better at reducing its footprint, but it needs to do more to get it into these small devices.

(g) **Multimedia support.**

DCE should support multimedia data streams. The following standards may apply:

(i) H.320 (Narrowband video teleconferencing (VTC)).

(ii) H.324 (Low bitrate VTC).

(iii) H.322 (ISO-Ethernet VTC).

(iv) H.323 (Ethernet VTC).

(v) H.321 (ATM VTC).

(vi) H.310 (High Resolution ATM VTC).

(vii) T.120 (Multimedia Data Conferencing (file transfer, still image transfer, shared white board, all in a multipoint conference)).

(h) **Internet protocol version 6.**

Multimedia streams included encoded video, audio, image, and computer application data.

This requirement has not been reviewed by anyone, so consider it a topic for discussion as opposed to a firm requirement. Perhaps the capabilities already present in DCE pipes and DCE Web can handle the multimedia needs of most DCE users, except for conferencing.

15. BRANDING AND METRICS

Some effort is being made to provide an XDCE branding program for DCE within the X/Open branch of the Open Group. This is an excellent idea, with the following cautions.

No DCE implementation should be allowed simply to purchase an XDCE brand prior to successful completion of a DCE validation test suite. The Cell Exerciser work done to date may serve as a foundation on which to complete a validation test suite.

The XDCE brand should only be granted to an implementation in its entirety. There should be no brand for just implementing the RPC, for example.

16. WORK UNFINISHED

Work on specifying the following requirements is yet to be assigned:

(a) Environment.

(i) Real-time support.

(b) Attributes.

(i) Brokered services.

(ii) Registry database(s).

(iii) Generic services API.

17. ITEMS TO BE ASSIGNED TO A SECTION

The following items are still yet to be classified and categorized within this document:

(a) Pipes.

Extend pipes, or add a new one, so you do an RPC for ‘‘session control’’ and then get multi-media pipes for sending video, etc.

18. ACKNOWLEDGEMENTS

Individuals and entities contributing requirements to this RFC:

(a) Len Swatski, Defense Information Systems Agency.

(b) John Dohm, Deloitte & Touche LLP.

(c) Dave Skeen, LEXIS-NEXIS, 9443 Springboro Pike, Miamisburg, OH 45342.

(d) Harold W. Lockhart Jr., Locus Computing Corporation (now Platinum).

(e) Chris Cowan, ISSC.

(f) Gary Gaskell, Distributed Systems Technology, University of Technology, Brisbane, Australia.

(g) Doug Brown, Sandia National Labs.

(h) Steve Latz, J.P. Morgan (now at IntelliSoft).

(i) Jonathan Chinitz, IntelliSoft Corp.

(j) Michael Guidry, Phillips Petroleum Company.

(k) Mark Michael, Hughes Electronics.

(l) William A. Estrem, 3M.

(m) Doug Eltoft, University of Iowa.

(n) Dave King, Barclays Bank.

(o) Tony Carrato, Hong Kong Jockey Club.

(p) Dr. Harald von Fellenberg, Union Bank of Switzerland.

(q) Anne H. Anderson, Hewlett-Packard Company.

(r) Ron Bjornseth, (303)977-2570, bjornset@ipsmlsron.den.mmc.com.

(s) Sue Ericksen, U S WEST Communications.

(t) Bill Sommerfeld, Hewlett Packard.

(u) Steve Jenkins, Jet Propulsion Laboratory.

(v) Ken Renard, Advanced Development Team, Army Research Lab.

(w) ‘‘Exchange’’ <<Need help identifing title of proceeding>>, Proceedings of the First ACM Conference on Computer and Communications Security November, 1993.

AUTHORS' ADDRESSES

Michael Guidry
Philips Petroleum
463 Information Center
Bartlesville, OK 74004
USA
Internet email: mguidry@bvemx.ppco.com
Telephone: +1-918-661-4384
Fax: +1-918-661-3720

Mark O. Michael
Hughes Electronics
PO Box 92919, Mail Stop SC S50/X366
Los Angeles, CA 90009
USA
Internet email: mmichael@igate1.hac.com
Telephone: +1-310-364-6759
Fax: +1-310-416-6181

William A. Estrem
3M Corporation
Information Architecture Dept.
St. Paul, MN 55144-1000
USA
Internet email: waestrem@mmm.com
Telephone: +1-612-736-6853
Fax: +1-612-733-3502