# NAC POSITION PAPER

## INTEROPERABILITY: A NAC POSITION PAPER

## ABOUT NAC

The Network Applications Consortium (NAC) is a strategic organization dedicated to increasing the availability and quality of applications for enterprise-wide computing by:

- publishing papers that state NAC's strategic vision of the industry's direction;
- educating vendors about enterprise-wide computing requirements;
- promoting, facilitating, and documenting collaboration among NAC members;
- advising distributed computing vendors on corporate planning and product development policies.

With these goals in mind, NAC is publishing its position on interoperability to help foster productive discussion among vendors and users. The paper discusses interoperability in the context of large, heterogeneous enterprises such as those that comprise the information technology infrastructure of NAC member companies. Members include:

*ABB Power T&D Co.*
*Arizona Public Service Company*
*Banyan Systems Inc.*
*Carolina Power & Light Co.*
*Conner Peripherals*
*Intel Corporation*
*Martin Marietta Energy Systems*
*NYNEX*
*Pacific Gas & Electric*
*Public Service Electric & Gas*
*Sprint Corporation*
*Tektronix, Inc.*
*University of Michigan*
*American Bureau of Shipping*
*Australian Bureau of Statistics*
*Bell Atlantic Mobile Systems, Inc.*
*Compaq Computer Corporation*
*Continental Grain Company*
*International Finance Corp, World Bank*
*MCI Telecommunications*
*Nike, Inc.*
*Pennsylvania Blue Shield*
*Resolution Trust Corporation*
*St. Jude Children's Research Hospital*
*Texaco*
*United States Marine Corps*

This paper is the result of NAC's Strategic Interest Group (SIG) process, a collaborative effort involving a subset of NAC members whose mission is to provide a cohesive NAC viewpoint on a particular industry sector or technical topic. The following NAC members were instrumental in preparing this position paper:

| | |
|---|---|
| *Continental Grain Company* | *Eric Dickstein* |
| | *Hilly Fuchs* |
| | *Bill Klauk* |
| *Martin Marietta Energy Systems* | *George Farris* |
| *MCI Telecommunications* | *Brian Plackis* |
| *NYNEX* | *Doug Savary* |
| *Pacific Gas & Electric* | *Art Beckman* |
| | *James Brentano* |
| *United States Marine Corps* | *Brad Triebwasser* |
| | *Mark Johnson* |

We welcome your feedback about this paper. For more information about NAC papers, contact:
Kelli Wiseth, Director, Technical Publications
Network Applications Consortium
c/o Pacific Gas & Electric Company
Mail Code B19A
P. O. Box 770000
San Francisco, CA 94177

# Contents

# Interoperability: A NAC Position Paper

## Executive Summary

Today's business environment is marked by rapid change and increased competition. In an effort to remain successful, corporate business managers look to the many available applications that promise to further the company business goals. At the same time, information technology managers are hamstrung in their efforts to deploy these applications because the building blocks comprising the foundation for applications don't interoperate.

The lack of interoperability limits the choices IT managers can offer as business solutions; given the lack of interoperability, IT managers must either:

- Choose the application that's best for the business, but which also requires different building blocks than what the infrastructure currently offers

- Choose the application from among those that will work with the building blocks that are already in place

Neither scenario is acceptable. Both scenarios impose limits on the IT manager's ability to provide the applications best-suited to supporting the business goals. The first scenario causes the company to lose its investment in the infrastructure, while the second scenario leverages the company's investment in the infrastructure, but at the cost of the best means to support the business goals.

On the other hand, if technology infrastructure components were interoperable, all applications would be available to meet business needs.

What is interoperability? To the IT manager, it is the ability to mix-and-match building block components and applications that comprise the IT infrastructure. In addition, components must use common services available on the enterprise. Thus, interoperability enables organizations to mix-and-match components — hardware platforms, operating systems, and applications — without loss of functionality or duplication of services: any client works with any service, and both use common services.

The ability to mix-and-match components and common services can only occur when all the pieces adhere to the same standards. Specifically, standard interfaces or application programming interfaces (APIs) are required between each of the building-block components. Only a small set of standard APIs is needed to connect applications to the services that they use. The industry must evolve to the point where all vendors use the same API sets.

NAC defines an effective marketplace process for developing standard APIs as follows:

1. The marketplace defines the need for a standard set of functions at a given interface point required to connect the components to each other and to the services they need.

2. In the early stages, vendors develop robust and competing APIs that effectively provide the functionality.

3. The marketplace declares an API winner through the most appropriate process model for the given market segment.

4. Winning API migrates to public ownership to facilitate adoption by all competing vendors as a standard.

5. All vendors continue to evolve the standard APIs as needs develop for greater functionality.

The benefits to the user community are obvious: standard APIs enable interoperability to occur; interoperability provides IT managers a choice — the choice of applications; choice of platforms; choice of operating systems; choice of networks — designed to best meet business needs without redundant support and maintenance costs.

As a consortium of large-site customers, NAC can serve an invaluable role in this marketplace process and make a significant contribution to the dialogue. Through papers such as this, NAC can sharpen the focus of its constituency on the relevant market and technical issues. By clarifying the customer issues, NAC can present a considered position representing a cross-section of industrial customers and act as a catalyst for evolution.

NAC has designed a six-step evaluation process for use in focusing its constituency, clarifying the issues, and developing its position. NAC will apply this process to the critical common services in the enterprise environment, including messaging, directory, and data access, to name a few. The outcome of such explorations will be the subject of future NAC position papers that will be presented to vendors and commercial users in the interests of fueling further dialog.

## Introduction

**Today's business environment is marked by rapid change and increased competition. To remain successful, corporate business managers look to the many available applications that promise to further the company business goals. At the same time, information technology managers are hamstrung in their efforts to deploy these applications because the building blocks that comprise the foundation for applications don't interoperate.**

"Do more with less." Virtually every company in the country has adopted this mantra for the 1990s. Around the globe, corporations are re-engineering, downsizing, or rightsizing as they attempt to remain profitable in an economic environment that is arguably the most challenging the world has seen in decades.

For many corporations, the use of information technology is what enables the "doing more." Properly harnessed, information technology can support and advance the business goals. As the technology marketplace continues its rapid evolution, business managers find no shortage of applications capable of meeting their specific business needs.

While there may be numerous solutions available, information technology (IT) managers find it increasingly difficult to implement many of them. One reason for this difficulty is that the organization's *infrastructure* — operating systems, hardware platforms, and networks — is composed of disparate elements that follow different standards, and thus, don't work together to support an application. Many of these building blocks can't be mixed-and-matched — they aren't *interoperable*.

Another difficulty arises due to the sheer number of building blocks from which to choose. New strategies and standards that define these building blocks surface each year, sometimes never to be heard from again, sometimes resurfacing later under different guises on different development paths. From AOCE, to MAPI, to VIM, to WOSA — with a raft of three- and four-letter acronyms in between — choices abound, whether they're vendor-driven or vendor-independent, de facto, de jure, or completely non-standard.

In addition to issues of short-term compatibility, senior IT managers wrestling with long-term strategic issues must also continually ask themselves: *Am I choosing the IT-equivalent of the Beta video format? Is this choice going to lock me in to a technological dead-end? Will the applications installed continue to work? When will all these pieces interoperate — work together — in the seamless fashion promised?*

This paper discusses these issues in the context of the enterprise-wide computing environment, and provides an overview of interoperability. The paper also

proposes a new process model for the marketplace. NAC has developed an analytical process whose intent is to provide all industry participants — vendors, developers, and industrial consumers — with a significant amount of in-depth data to help further productive discussion.

## The Enterprise-Computing Environment

**The enterprise environment is composed of disparate hardware platforms, operating systems, data types, and locations: a multi-vendor, multi-platform, distributed environment.**

Today's business model has evolved from a slow-moving, rigid hierarchy to a flat, modular organizational structure where flexibility and speed are essential. Likewise, the information technology environment has evolved to support the business goals by providing a flexible, responsive infrastructure that fits the best tool to the task. The result is a heterogeneous, multi-vendor, multi-platform operation, with a wide variety of installed hardware and software.

For example, the typical NAC member company supports virtually every desktop computing platform available on the market today, including Intel-based platforms running DOS, Windows, OS/2, and Unix; Macintosh platforms; and Unix workstations running any one of several versions of the Unix operating system. Many organizations also still rely on dedicated 3270 terminals connected to mainframes, or are early adopters of new technologies such as hand-held computers or PDAs (personal digital assistants) with pen-based operating systems.

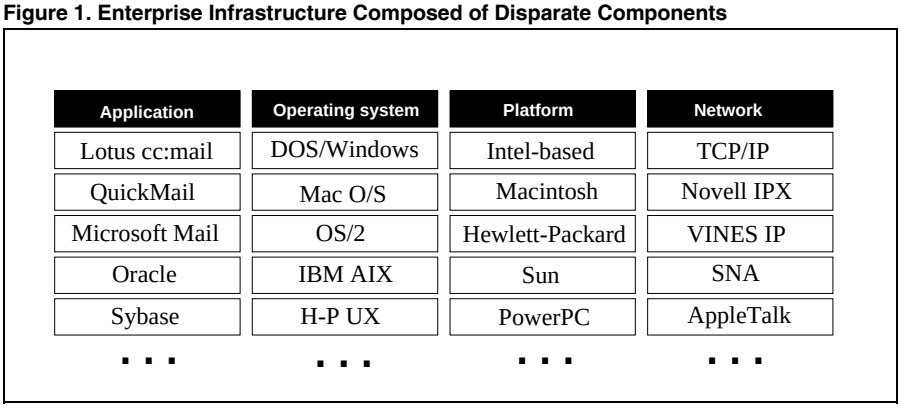
And that's just on the desktop. Corporate data is housed on legacy mainframe systems, minicomputers, file servers, and application servers, all running a variety of operating systems, from MVS, VMS, VM, OS/2, Windows NT, to Unix in all its flavors. The data format is equally diverse; today's enterprise network supports graphics, voice, and video, in addition to text.

Not only are the platforms, operating systems, and data types diverse, but the networks that shuttle data from one end of the enterprise to the other run the gamut from Banyan VINES, to Novell NetWare, to IBM SNA, to name a few. In addition, the business environment is no longer contained within four walls or within an eight-hour day, as business travelers on the road log in and work from hotels, airports, restaurants, the beach, or the front seat of their cars while sitting in traffic.

Figure 1 (below) provides a snapshot of some of the basic components described above that might comprise a typical company's IT infrastructure. While the list is by no means exhaustive, it does begin to demonstrate the abundance of different

types of building blocks available. From such a list, the IT manager must pick and choose technology that will best support the business goals.

**Figure 1. Enterprise Infrastructure Composed of Disparate Components**




## MIX-AND-MATCH CAPABILITY MEANS LEVERAGING INVESTMENTS

**The ability to mix components within the infrastructure — without loss of functionality — can bring the benefits of applications to the information technology infrastructure, and at the same time, save corporate dollars.**

The information technology manager's bottom-line goal is to support and advance the business goals by selecting the most appropriate applications from among the wide range of choices on the market today. Implicit in this concept is the premise that the "most appropriate" application in a given context may not be the most appropriate in another context.

A concomitant goal to delivering the "most appropriate" application is delivering that application at an affordable cost. Application costs include not only direct expense — purchase price of the application, the hardware platform, or the operating system, for example — but also the indirect costs — installation, training, support, and maintenance — that are attendant with any new application. Thus, the most appropriate choice between two applications may be the one with the higher initial acquisition cost but with lower training, support, and maintenance costs.

This fundamental IT mission — deliver the most appropriate application from among the many choices (see Figure 1 above) and, at the same time, drive down operating costs — necessitates the ability to "mix-and-match" from among the choices. That is, in order to keep both direct and indirect costs down, organizations must be able to choose freely from among available business applications and all the other building blocks. Being able to implement applications on the organization's chosen hardware platforms, operating systems,

and networks means the company can capitalize on the existing investments without incurring additional costs for support, training, and so forth.

Conversely, if an organization must adopt new or different building blocks in order to implement a chosen business application, the organization is saddled with increased costs at every step of the way, from implementation through on-going maintenance. For example:

- acquisition costs are higher because splitting dollars among several platforms doesn't provide the company appropriate leverage with vendors for the most favorable quantity discounts

- implementation costs are higher because technical staff must be trained on another hardware platform, operating system, or network, instead of what's already installed

- support costs are higher; employees with the appropriate skills must be available to support the application, so existing employees must be re-trained to support another hardware platform, operating system, or network

- training costs accrue for end-users, support personnel, and developers

In short, companies must apply technology to their businesses at an affordable cost — affordable for the original installation, and affordable from the standpoint of additional training, ongoing support, and maintenance. And *interoperability* — the ability to mix-and-match components — enables companies to leverage all these investments.

## Interoperability

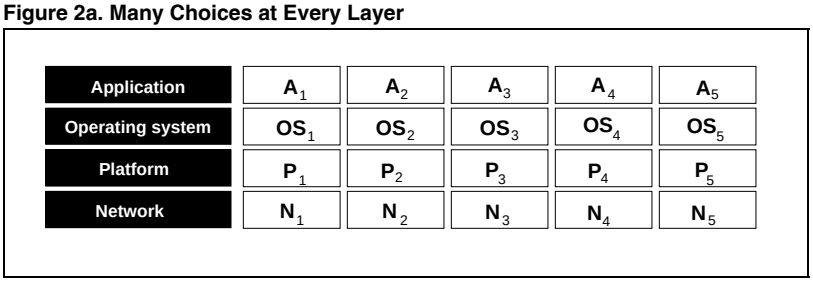
**Interoperability enables organizations to mix-and-match components — hardware platforms, operating systems, and applications — without loss of functionality or duplication of services: any client works with any server, and both use common enterprise services.**

From a non-technical, business perspective, the concept of "interoperability" enables an organization to leverage investments made in the building blocks that comprise its IT infrastructure. As described earlier and highlighted by Figure 1, these building blocks comprise a diverse group of networks, platforms, operating systems, and the applications that run on them. Another conceptual view of these components is shown in Figure 2a below. (Note that this is also an oversimplified view; the enterprise environment might include different layers, or the layers might be organized differently. For simplicity's sake, the example a base upon which to build a definition of interoperability.)

Saying that all these components should *interoperate* has shades of meaning that depend upon your vantage point. For example, in an individual client workstation, an application, such as a word processor, must be interoperate only with the given operating system; in turn, the operating system must run on the hardware platform in question. In addition, if the workstation must access a service over the network (and is thereby acting as a client), the network layer must be compatible with all these components.

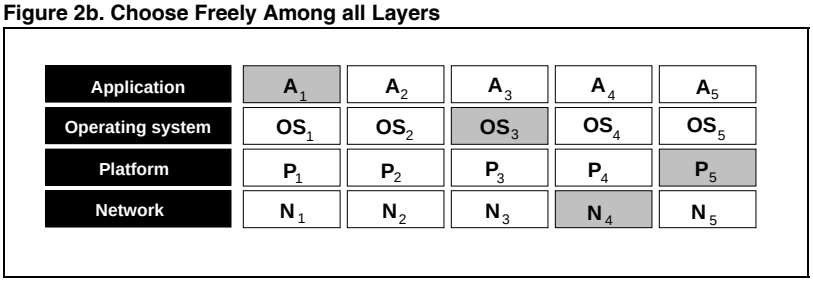
Implementing a single application, such as a word processor, on a single hardware platform or a particular operating system is one thing. But, as shown in Figure 2a below, in the process of implementing enterprise applications, one discovers a great number of choices exist at every layer of the model:

**Figure 2a. Many Choices at Every Layer**




Interoperability would enable IT managers to choose any component from within any layer without loss of functionality. For example, after selecting any application[1] — $A_1$ for example — you may also choose $OS_3$, $P_5$, and $N_4$ as the other components of choice with the assurance that all will work together.

**Figure 2b. Choose Freely Among all Layers**

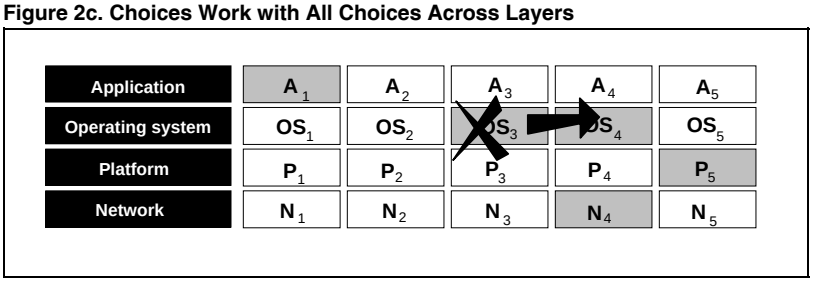



Interoperability exists when choices can be made from any layer and matched with choices in any other layer. So if a decision were made to move from one operating system to another — for example, from $OS_3$ to $OS_4$ as in Figure 2c below — the operating system layer would be interoperable with the application and platform layers if the change could be made without affecting either of these

[1] Note that the *application* can be either the client or server application code.

layers. For example, the e-mail client application would still run on the selected hardware platform and the newly chosen operating system, and the network operating system would still deliver messages.

**Figure 2c. Choices Work with All Choices Across Layers**




Moving beyond a single workstation to the enterprise at large, interoperability would enable an organization to implement a given hardware platform and e-mail client application for one department, and select a completely different hardware platform and different e-mail client application for another department with complete assurance that both e-mail clients could communicate with each other.

However, being able to mix-and-match components and applications isn't enough. Any enterprise application requires certain basic functionality, including the ability to secure the application so that only the appropriate parties can access it, the ability to locate the application on the network, and the ability to transmit information from the starting point to the appropriate endpoint. These *common services* — security, directory, and messaging, to name a few — are required by many enterprise applications.

Therefore, in order for the environment to be truly interoperable, organizations must be able not only to mix-and-match building blocks on any level, but the basic functions — common services — that provide the foundation for the enterprise environment should be seamlessly used by all applications. Not doing so means duplicate services, schemes to connect and synchronize multiple systems, additional administration, excessive training and support costs, and so on.

Thus, NAC's concept of *interoperability* has two dimensions. Interoperability provides IT managers with

- the ability to mix-and-match the building-block components and applications that comprise the IT infrastructure
- the use of a common set of service functions shared by all applications

The following two scenarios describe in functional terms what occurs when infrastructure components are interoperable — and when they aren't.

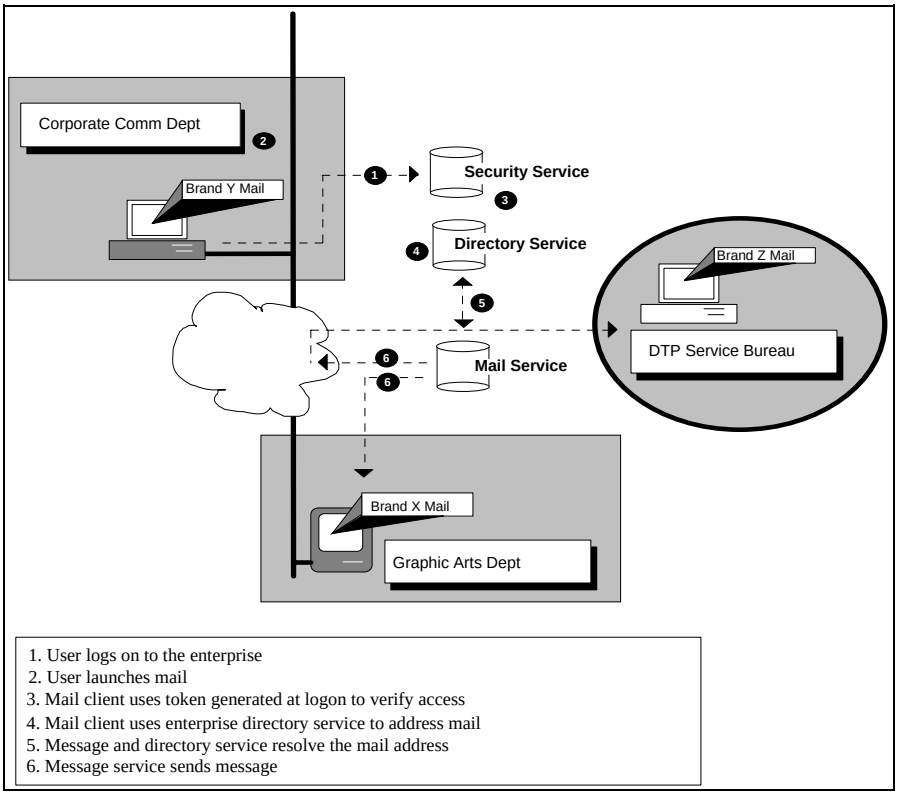
## INTEROPERABILITY FROM AN END-USER PERSPECTIVE: THE IDEAL

**Interoperability at its best provides end-users with a productive work environment that supports their needs at every step of the way. They have seamless, transparent access to the information, applications, services, and the other end-users they need to get their jobs done.**

Late one Friday afternoon, a writer in the corporate communications department discovers he needs the corporate logo, in electronic form, to insert in a brochure that's due at a desktop-publishing service bureau in a few hours. He's already logged on to the network, so he opens his mail application and begins browsing the corporate directory. Since he doesn't know anyone in the graphic arts department, he searches the electronic directory, by job title, until he locates the lead designer. Double-clicking on the directory listing pops her full e-mail address into the mail message. He quickly writes a note explaining his situation, marks the message *urgent* and *return receipt requested*, and sends it on its way.

Behind the scenes, the messaging service uses the enterprise directory service to route the message to the designer. Although the lead designer uses a different mail application and has different hardware with a different operating system, the incoming urgent message is correctly moved to the top of her incoming mail queue, and an alert beep sounds on her Mac. As she opens the message and reads the writer's request, the writer receives the *receipt* message.

**Figure 3. Interoperability: The Ideal (An Example)**




After the designer reads the message onscreen, she clicks the *Reply* button. She picks up the microphone connected to her computer, rattles off a few instructions to the writer about how to use the logo file, and sends both the audio file and the scanned logo image file as her reply.

An alert beeps on the writer's computer when the reply comes in, and he opens the message. Double-clicking on the audio icon, he listens to the instructions from the designer. He's never used this particular feature in his mail package — in fact, he didn't even realize it had this feature — and he makes a mental note to look into how to create notes like this himself, when he has more time. He detaches the logo file, opens the document in his word processing application, and then places the logo itself — converted automatically to the appropriate graphic format by the word processor — in position in the brochure.

After proofing the brochure one last time, he composes a note to the service bureau, attaches his brochure document file, and sends the e-mail message to the service bureau. Although the service bureau uses a different e-mail system on an entirely different network, the message is received, and the files are downloaded and printed to the high-resolution laser printer without a hitch.

* * *

Note that in this example:

- Applications, operating systems, platforms, and networks have all been chosen on the basis of fitting the best tool to the task.

- Applications work seamlessly in the enterprise environment because they use a base set of common functions, provided by common services.

## INTEROPERABILITY FROM AN END-USER PERSPECTIVE: THE REALITY

**Reality isn't pretty — nor is it efficient or cost-effective. Corporations can't mix-and-match components without paying a price, whether it's lost functionality, poor performance, increased training costs, excessive administrative burden, or duplication of services.**

Unlike the seamless scenario described above, you're more likely to see something like this: On a Friday afternoon a designer in the graphic arts department sends a writer in the corporate communications department an e-mail message containing a scanned image of the company logo to use in a brochure. Come Monday morning, the writer calls the graphic arts department looking for the logo. Turns out the graphic arts department's directory entry for the writer includes a middle initial — but the corporate communications directory doesn't — so the message never got through.

The designer then re-addresses the message, typing in the writer's correct address by hand instead of using the e-mail directory. Seconds later, the message bounces back to the designer's in-box: "Format not supported." Turns out the designer had also attached to the e-mail message an audio instruction file — designers hate to write — a type of attachment not supported by the box that connects the two mail systems together. The designer deletes the audio message, types in a few words instructing the writer how to insert the file into his brochure, and sends the message again, crossing her fingers.

Meanwhile, the writer does what he always does when he has "computer problems:" He reboots his PC. The startup screen appears, he logs onto the network, downloads his document from the file server, and begins making some edits to the brochure while he waits for the logo.

After about 15 minutes, he snags the network administrator, on his way to solve another problem. "I've been waiting for some mail from the graphic arts department for awhile now... is something wrong with the network?" The system administrator scans the writer's desktop for a few seconds and then tells the writer that he's not logged on to mail. "But I just rebooted and logged on."

The system administrator tries to explain the philosophical differences between logging onto “the network” and logging onto mail, and why both are necessary. The writer shrugs, launches his mail application, and logs on to mail.

The message finally appears in the writer’s in-box. An hour later, the writer calls the designer. “You know that file you sent me? I can’t open the darn thing on my machine.” Turns out the graphic arts department is on Macs and the corporate communications department is on PCs. The designer uses a Mac-based utility to export the file to a DOS disk, throws on her Nike’s, and sprints — it’s begun to rain — across the corporate campus to the communications department.

That weekend, the network system administrator spends 12 hours synchronizing the various mail directories throughout the system.

* * *

Such a scenario is not only frustrating, it is also fairly typical, and represents lost productivity.

## The Need for Standards

**The ideal — mix-and-matchable components supported by common services — occurs when all the pieces adhere to the same *standards*.**

The difference between "the ideal" and "the reality" scenarios above is easy to explain. Transparent interoperability — the ability to mix-and-match components and a set of common services — can only occur when all the pieces use the same *standards.* Because many components on the market follow different standards — or because a "standard" implementation has been modified and is thus made essentially non-standard — interoperability is difficult to attain in today's distributed, client/server environment.[2]

Common buzzwords like *distributed environment* and *client/server* belie the underlying complexity of the concepts they describe. Key to these concepts is a splitting of the processing functions into two (or more) pieces, each of which plays a different role. The client submits tasks to the server, which performs the processing, in some cases returning values to the client, in other cases performing another task.

For example, in a mail system the client application might submit messages to the mail server, which in turn processes the message and initiates a notification process to alert the mail clients to which the message was sent.

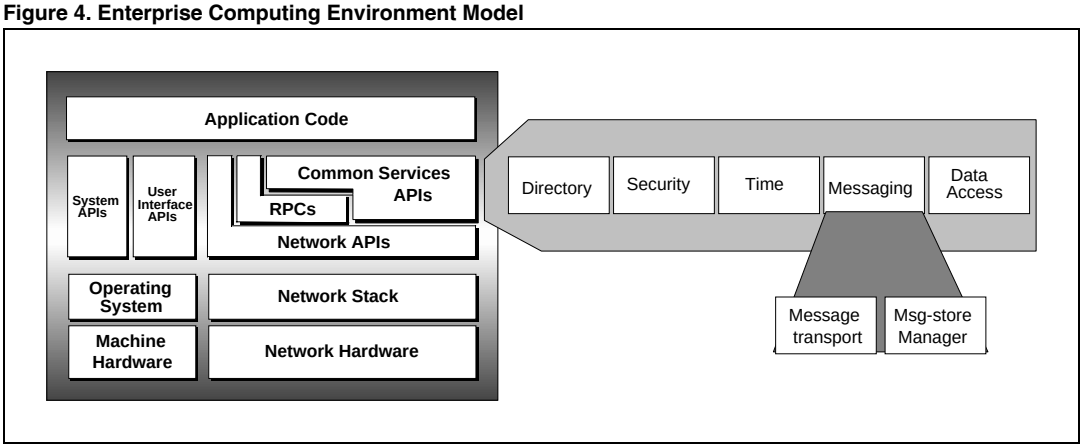
The mechanism used between the pieces of the application — between the client process and the server process — is the *application programming interface* (API). The client application uses the language of the server application's API set — makes calls using functions from the server's API library — to access a function provided by the server.

Going beyond the simplified view of the enterprise building blocks (Figures 2a, 2b, and 2c), Figure 4 below presents a high-level view of the key interface points between the building blocks in a typical enterprise. All the basic components that make up the enterprise computing environment can fit within this model.

---

[2]Different systems can usually be connected with *gateways*, which translate between different databases, mail systems, or networks. But gateways typically provide a least-common-denominator solution.

**Figure 4. Enterprise Computing Environment Model**




For example, the *Application Code* box in the model would contain desktop applications like Word, Excel, or WordPerfect, as well as client database applications and client mail applications. Applications in this box must make calls using APIs available in the *User Interface APIs* box. In the example described, the APIs could be to the Windows environment or to the Mac operating system.

Many models exist in the marketplace today for addressing the need for standards at some or all of these interface points. However, none of these models provides a complete solution:

- OSF's DCE (Open Software Foundation's Distributed Computing Environment) provides a means of developing applications and provides services such as directory, security, and time, but doesn't provide for messaging or data access. Nor is there significant support currently in DCE for the legacy desktop systems already in place; that is, Windows, DOS, and Macintosh.

- WOSA (Windows Open Systems Architecture), on the other hand, supports the Windows desktop with specifications for messaging and data access APIs (MAPI and ODBC, respectively). But WOSA doesn't include security or directory services as yet.

- COSE (Common Open System Environment) and (Common Desktop Environment) provide a solution, but only in the Unix arena.

- IBM's Open Blueprint fills all the boxes in the model, top-to-bottom — but with mostly an IBM-centric view.

- AOCE (Apple's Open Collaborative Environment) only collaborates between the messaging models of other AOCE-supported products, and at the moment, not that many exist.

- Solaris' Federated Services, similar to Microsoft's WOSA, consists of a network services architecture built beneath a set of application interfaces provided by the operating system.

- Novell AppWare, an application-development environment NLM (NetWare Loadable Module) that creates object-oriented applications in the NetWare environment.

In addition to being incomplete, these approaches begin to highlight another key issue: for any given interface point in the model, there are simply too many choices. The market provides too many competing standard APIs. For example, the application-to-directory interface point on the model above has several alternatives on the market today, including X.500, VINES StreetTalk, Novell NetWare Global Directory, DCE's CDS (Cell Directory Service), DCE's GDS (Global Directory Service), to name just a few.

What's needed is only a small set of standard APIs that accomplish the simple yet essential task of connecting applications to the services that they use. The industry must evolve to the point where all vendors use the same API sets. The ideal is one standard API that will evolve over time as the need for newer functionality is required.

## The NAC Marketplace Process for Selecting a Standard

**NAC proposes taking the best of existing standards processes and developing a new process of competition that will select a single, standard API for use by all vendors.**

Examining the list of competing models and APIs (above) reveals several aspects of the technology marketplace. For starters, the "marketplace" is not a homogeneous mass but is segmented roughly along the lines of the different layers depicted in Figure 2a. In addition, the different layers of the marketplace are at various stages of maturity in terms of standards development and market penetration. For example, the network layer is fairly mature, with entrenched standards like Ethernet and TCP/IP; no new vendors are emerging trying to ply new low-level or transport protocols; there'd be no point.

Likewise, different standards processes, with their own relative merits, are at work in the technology marketplace. For example:

The ***technical-merit (open) process*** — best typified by the IETF (Internet Engineering Task Force) — is driven by the need for technology, not by market pressures. The IETF's TCP/IP protocol suite is the de facto standard for wide-area networking today because it was open — the spec was published and available to the public, so any company could use it — and it worked. However, the IETF standards process is research, not business oriented, and thus can't directly respond to business market pressures.

On the other hand, a process totally driven by market pressures can be described as the ***market-leader process***. IBM's SNA (Systems Network Architecture) mainframe protocol, which has no competitors, is an example of a standard that dominates due to lack of competition and exclusive ownership within its market segment.

The ***committee process,*** best exemplified by the work of ISO (International Organization for Standardization), is an example of a process that attempts to anticipate the marketplace rather than react to it. The downside of this process is that standard definition and product delivery are too far apart to satisfy the needs of the marketplace.

Recently, a more ***pragmatic process*** has emerged, as seen in the work of COSE (Common Open Systems Environment) on the Common API Specification (aka, Spec 1170). Virtually all Unix vendors got together to develop a single, supersetted version of Unix that will enable interoperability among all existing versions. The benefits of this approach are that the specification is coming to market quickly, it's backwards-compatible to existing systems, and clear guidelines will be available for all for any further development.

Of course, NAC would like to avoid the type of situation that necessitated the development of Spec 1170 in the first place — the multitude of incompatible, competing Unix versions. Nonetheless, the process undertaken to develop Spec 1170 represents a significant step forward for the industry. Faced with the fact that maintaining different versions of applications to account for the nuances of the variety of Unix versions available, vendors got together to solve a common problem, recognizing that, in the long-run, all would benefit.

NAC encourages such activities, and believes that an effective marketplace process can include the multi-vendor participation and sponsorship of the pragmatic process, combined with the openness of the technical-merit process, tempered by the realities of the market-leader process. Thus, NAC's marketplace process model is described as follows:

1. The marketplace defines the need for a standard set of functions at a given interface point.

2. In the early stages, vendors develop robust and competing APIs that effectively provide the functionality.

3. The marketplace declares an API winner through the most appropriate marketplace model for the given market segment.

4. Winning API migrates to public ownership to facilitate adoption by all competing vendors as a standard.

5. All vendors continue to evolve the standard APIs as needs develop for greater functionality.

Behind this process are several important concepts:

- the sooner step three is achieved, the sooner greater benefit will be realized by both the vendor and the user communities

- success can only be achieved with this process when vendors relinquish ownership of an API so that the rest of the industry will adopt it for use

The benefits to the user community are obvious: standard APIs enable interoperability to occur; interoperability provides IT managers a choice — the choice of applications; choice of platforms; choice of operating systems; choice of networks — designed to best meet business needs without redundant support and maintenance costs, discussed earlier in this paper.

The vendor community will realize bottom-line benefits as well. By adopting a common set of standard APIs, a vendor's product will be among the choices when it comes time to selecting applications and components. Without standard APIs, vendors lose opportunities for new and ongoing revenue when their products aren't among the choices being evaluated by the business community.

For example, a large corporation with 20,000 internetworked PCs has a mail backbone consisting Banyan VINES Intelligent Messaging. When the corporation decides to select a new mail client, the lack of standard APIs limits the choices down a small number. Many vendors who do not support Banyan's specific APIs lose the opportunity for the corporation's business, and lose the opportunity for revenue on 20,000 seats.

Another benefit vendors will realize by adopting this process is improved time-to-market. By adopting a limited-size set of APIs that use common services — instead of providing those services in each of their products — vendors should realize a considerable savings in development time, which in turn leads to competitive advantage.

* * *

As a consortium of large-site customers, NAC can serve an invaluable role in this marketplace process and make a significant contribution to the dialogue. Through papers such as this, NAC can sharpen the focus of its constituency on the relevant market and technical issues. By clarifying the customer issues, NAC can present a considered position representing a cross-section of industrial customers and act as a catalyst for evolution. NAC will use a six-step analysis and recommendation process to derive its position.

## Analysis and Recommendation Process

**By applying a six-step evaluation process to a particular service or application area, NAC will be able to present a considered viewpoint, with recommendations for both vendors and users regarding the selection of particular APIs.**

An in-depth evaluation process, the six-step process will enable NAC to:

- present a coherent view of the marketplace, the players, and the issues
- evaluate the marketplace on the basis of market penetration
- unearth the technical foundations underlying the issues
- evaluate different approaches on the basis of technical completeness
- provide vendors with considered statements of strategic direction

The process is described below. As a starting point, NAC will propose a generic model of the application or service under scrutiny. The generic model will be based on typical functionality available in most products on the market, with the understanding that a generic model cannot possibly capture all models.

1. Design a generic version of the application or service. Summarize the key presumptions about the service, its components, and its basic functionality by defining:
   — an architectural model
   — the underlying structure

2. Based on the generic application or service, identify and define the key components that must interoperate.

3. Extrapolate the interface points by displaying a matrix of rows and columns that represent the interface points as requestors or providers of functions. Develop a list of the functions that must occur between all interface points to enable a full-featured version of the generic model proposed in the first step above.

4. Compare the interface point matrix with the functional listing and group the interface points into functionally similar API sets.

5. Map the competing standard APIs, formats, and key industry players to the API sets defined in the step above.

6. Evaluate existing standard APIs on the basis of market penetration, technical completeness, market segmentation, and openness. Develop recommendations for vendors and consumers for each API set.

NAC believes that only by following this detailed a process can vendors and consumers have more meaningful discussions regarding interoperability.

## Conclusion

On the one hand, *interoperability* is easy enough to define. To the IT manager, it is the ability to *mix-and-match the building block components and applications that comprise the IT infrastructure*. On the other hand, any exploration of the "building blocks" that comprise the infrastructure reveals a rather cluttered playing field composed of disparate elements driven by different market and technical influences.

To more fully explore the issues surrounding a particular segment of the field — e-mail systems, directory services, security services, for example — NAC will apply its six-step analytical process and publish its findings in future papers. NAC believes that this process will provide a new, deeper level of understanding of the issues, and that such depth will fuel a more productive discussion among vendors and users.

## References


Banyan Systems Incorporated. *VINES Mail Client Programming Interface*.

Burton, Craig and Lewis, Jamie. *Messaging Interfaces: MAPI, VIM, XAPIA/CMC, Netware SMF71*. Burton Group Report. October 1993.

*Client Developer's Guide: Messaging Application Program Interface Software Development Kit, Version 1.09*. Microsoft Corporation. 1993

*Developer's Guide: Common Messaging Call Application Programming Interface Version 1.09*. Microsoft Corporation. 1993.

Hammer, Michael; and Champy, James. *Reengineering the Corporation*. HarperBusiness, A Division of HarperCollins Publishers. New York, 1993

Malamud, Carl. *Analyzing Novell Networks*. Van Nostrand Reinhold, New York, 1990.

Mercilliot, Marc, Carr, Eric, and Anderson, Ron. *Why is E-Mail Still So Bad?* Network Computing. May 1 1994.

Morse, Stephen. *Microsoft Mail Directory Synchronization*. Network Computing. January 15, 1994.

*Netware Global MHS and Netware 4.0*. Novell Inc. 1993.

Rosenberry, Ward; and Teague, Jim. *Distributing Applications across DCE and Windows NT*. O'Reilly and Associates, Inc. Sebastopol, California, 1993.

Rosenberry, Ward; Kenney, David; and Fisher, Gerry. *Understanding DCE*. O'Reilly and Associates, Inc. Sebastopol, California, 1992.

*Spec 1170 Explained and Analyzed*. Uniforum Monthly. April 1994