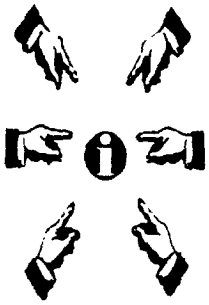

Microsoft® Windows™ Cairo

Product Planning

Product Requirements

File: prd7ext.doc

Advanced Systems Business Unit Program Management

Version 7

March 8, 1993

Distribution: Limited to Microsoft Advanced Systems

Microsoft Confidential

Printed on 3/8/93 at 11:40 AM

Note: This is a product planning document for a software system that is still in development. Some of the information in this documentation may be inaccurate or may not be an accurate representation of the functionality of the final retail product. Microsoft assumes no responsibility for any damages that might occur either directly or indirectly from these inaccuracies.

Contents

Windows Cairo Product Requirements

Important Notice: This is a draft distributed to the Microsoft Systems Division only. Please do not distribute this to anyone else.

This document describes the goals of the Windows Cairo product, what is in the product, and what is not in the product. It defines the product for the benefit of various stakeholders in the product.

I. Product Overview - Greg Lobdell

Windows Cairo is the second generation of the Microsoft® Windows NT™ operating system. Cairo is the first Microsoft operating system to focus on delivering a platform which realizes the *Information at Your Fingertips* vision.

Cairo's Mission

Deliver the advanced Windows operating system for sharing information in a distributed environment

There are several important components of this mission:

- *Advanced Windows* Cairo is the *second* generation Windows NT -- wherever NT goes, Cairo goes (x86, MIPS, Alpha, etc.). And with Cairo, Windows NT will move into the mainstream corporate desktop market.
- *Operating System*: Cairo is first and foremost an operating system- delivering services to applications and users.
- *Information Sharing*: Information is only useful if it is shared -- sharing information is the key step we make in delivering IAYF. Cairo's information sharing functionality is compelling in and of itself.
- *Distributed Environment*: Cairo is being designed for the organization where multiple systems are connected on a LAN -- with scalability from 2 to 200,000 nodes.

Product Definition

In order to achieve this mission, Cairo will be delivered to the market in two principle packages. Packaging details are provided later in this document; everything that follows should be thought of as part of these two packages:

- *Cairo Advanced Desktop*: Complete operating system for all client desktop needs. Includes all components of Cairo with functionality focused on enabling information sharing in a distributed environment. To be useful on a network, the desktop system must be connected to at least one Cairo server.
- *Cairo Advanced Server*: A true superset of the Advanced Desktop package -- adding distributed network functionality and additional system administration tools and monitoring functionality.
- *Cairo Design Environment*: This package provides the design environment for authoring smart-folders. It includes the core development tools including Visual Basic for Applications environment/debugger and all documentation on the environment and Visual Basic for Applications language.

Target Market

Cairo's target market is corporate desktops and corporate servers requiring advanced operating system services in a distributed computing environment.

On going research (within MS Systems) to determine a task-based segmentation model of the operating system market has helped identify the following key segments for Cairo[1]:

- Inside Professionals: E.g., product manager, accountant, sales manager. They are huge information consumers.
- Mobile Professionals: E.g., Salesperson, consultant, auditor. Intermittently connected user. Segment has very high growth rate
- MIS/PC support and System administrators: E.g., PC Hardware Consultant, Network administrator. Highly influential in the adoption decision cycle
- MIS Solution Providers: E.g., Systems analyst, macro language or VB developer for custom solutions. Very influential segment in adoption decision cycle.
- ISVs/Corporate Developers: E.g., Hard-core application developers-- Aldus, American Airlines Sabre project (people who use C/C++, MASM, PDK/SDK as tools).
- Engineers: E.g., Chemist, Aeronautical engineer, Toxicologist - the traditional workstation market.

For simplicity, the remainder of this document will consolidate these segments into three categories:

- End-users: Inside Professionals, Mobile Professionals, Engineers
- MIS/System administrators: MIS/PC Support and System Administrators
- Developers: MIS Solution Providers and ISVs/Corporate Developers

User Experience and Benefits

When looking at the benefits and functionality of Cairo, it is important to remember that everyone is an end-user--they interact with the basic services (particularly the user interface) performing similar tasks. Clearly, MIS/System administrators go beyond the basic services and have specific requirements (as do developers), but everyone has to interact with the user interface and the services it exposes.

Windows Cairo will provide significant benefits to all three customer categories.

User Experience and Benefits

User Interface: Cairo users will see a user interface similar to Chicago with the following benefits:

- The UI desktop models the physical desktop used in every day business. Users experience Cairo by focusing on their work, rather than on the system or applications that perform work. Key to this is the reliance on direct manipulation -- dragging a document to a printer object causes the document to be printed. This consistent and expected behavior results in a shorter-learning curve, lower training costs, and an increased sense that the system is helping them get their jobs done.
- Special purpose managers (e.g., File Manager, Print Manager, Program Manager) are gone. Interaction with objects on the desktop is direct manipulation with heavy reliance on drag-and-drop actions.
- Behavior specific to an object is exposed by its context-menu (accessed through right mouse button) for consistent access to actions (e.g., open, edit...etc.).
- Objects on the desktop can be physically present, or references. With references, the object isn't physically on the desktop, but rather the icon represents a link to the actual object. Links track (invisibly to the user) as the object moves-- meaning users don't have to worry where the object moves to.

[1] We have consciously eliminated the home-consumer, corporate clerk, and corporate administrative support segments from our consideration set because the tasks they perform and hardware characteristics do not intersect with Cairo's feature-set and/or hardware requirements.

- The Explorer provides the user with intuitive and powerful ways to *explore* for information. Viewing and browsing data (often with customized views) assists the user get the information "at their fingertips".

User Interface: Cairo evolves beyond Chicago with the following:

- *Smart Folders* (a.k.a. Delphi collections): Users will see certain folders with "intelligence." That is, properties on the folder define both the way information is stored in a smart-folder and the attributes of that information. The Cairo User Assistance system (Help) is an example of smart-folders in use. In addition, Cairo will ship at least an Email client, Bulletin Board System, and Document Management system as useful examples of smart-folders. Basic operations with smart folders are identical to standard shell folders -- and most importantly, the user interface is through the Explorer.
- Finding information is much easier through queries. Cairo provides content-indexing for all popular applications (which produce any kind of document). This means finding information can be virtually instantaneous. For example, finding all documents containing "Novell Netware 4.0" is now a simple query, rather than a painful search through the directory hierarchy looking for clues in file names.
- The network disappears and users feel as if they have a huge pool of information and resources at their fingertips. The user's view of their machine and the network appears as a single namespace. Users no longer have to connect to explicit network shares to find information.
- User Assistance (Help): Users have 2 levels-- QuickInfo for simple pop-up help and detailed help articles which are browsed in the Explorer as a customized collection (and queried). Articles can be annotated by MIS departments or an individual (with appropriate privileges) to make the assistance relevant to the particular environment.
- Task Assistants allow users to easily automate routine tasks (such as filtering and filing email messages). The graphical environment for creating Task Assistant scripts is intuitive and easy to learn.
- Existing applications run: If the user has MS-DOS, Windows 3.x, Windows NT 3.1, or Chicago applications they run the same in Cairo.
- Existing applications can take advantage of Cairo services such as content indexing, smart-folders, and replication without changes. From the users perspective, the same applications now appear to offer additional benefit.

System administrators and MIS: In addition to the benefits described above,

- Distributed support that provides an infrastructure appropriate for "downsizing" enterprise networks onto Cairo-based, micro-processor servers. This infrastructure includes a distributed file system namespace, a directory service, replication, and security.
- Scalability- the distributed infrastructure has been designed to be easily scalable -- from small workgroups to enterprise-wide installations with hundreds-of-thousands of users.
- The Directory Service is tightly integrated into the system. The DS is part of the same namespace, has the same programming interface, same storage, same object model, uses the same properties/index/queries, and is replicated using the system replication services. It is inseparable from the system yet easily connects to foreign DSs.
- Easily add organization and structure to unstructured information. Cairo enables users to easily create powerful information organization and sharing 'applications'. These applications are really "smart folders" and are designed with the *Cairo Design Environment* (CDE). With the CDE they can add properties and behavior to objects and control the way the user interacts with the smart-folders.

  Bundled with Cairo will be at least the following smart-folders each designed to be used "as is", but more likely, to be customized to fit organization needs:

- Email client
- Bulletin Board system (replicated group memory featuring multiple topics, message threading, and read/unread tracking).
- Document Management system: document management -- check-in & check-out, and sequential versioning.

- Extensible Help system allows system administrators to annotate Cairo's help documents with information relevant to their environment.
- Simplified administration via remote system management; distributed security allowing single logon; a fully extensible user interface; and support for centralized or decentralized administration, depending on administrator preference and site requirements; and
- component-based development that allows system managers to quickly design and build applications.

Programmers: In addition to the benefits described above for end-users,

- an evolutionary path based on OLE 2.0 and Win32;
- Component software construction is made much easier with the *Component Object Runtime* (COR). COR enhances OLE2 by enabling *software connectors*-- providing a standard way for objects to describe their structure, binding characteristics, and persistent binding information.
- performance and storage improvements for OLE 2.0 applications with OFS-- the installable file system that supports persistent storage of OLE2 objects.
- the opportunity for applications to fully exploit fast queries on information that has been content indexed.
- native support for distributed computing services including remote activation, replication, directory service, and security.

Product Definition, Structure and Features

<table>
<tr><td colspan="4">User Environment: UI Shell, User Assistance, Task Assistants,
Content Based Query</td></tr>
<tr><td colspan="2">"Business Solution" Development Support:
Cairo Design Env. (Delphi), Components</td><td>InfoDoc</td><td>International
Language Support</td></tr>
<tr><td rowspan="2">Component Object
Runtime</td><td rowspan="2">System Management</td><td>Distributed File System</td><td rowspan="2">Content
Indexing</td></tr>
<tr><td>Replication</td></tr>
<tr><td colspan="2">Directory Service</td><td colspan="2">Access Control (Security)</td></tr>
<tr><td colspan="4">Distributed OLE (CairOLE)</td></tr>
<tr><td colspan="4">Object File System</td></tr>
<tr><td colspan="4">Windows NT 3.1</td></tr>
</table>

Figure 1. System Structure.

Windows NT

As the next generation of Windows NT, Windows Cairo builds upon the foundation provided in Windows NT 3.1. Many parts — such as the base operating system and device drivers — remain the same; the features not described in this document are similar to those in Windows NT. Areas where Cairo augments or replaces Windows NT functions, such as with the new Object File System, are noted.

User environment

Intuitive, Object-oriented User Interface. Users can take advantage of direct manipulation techniques (drag-drop, point-click) to handle a diverse set of objects straight from the desktop. Moreover, the user interface supports links to track objects in different locations.

Explorer. The Explorer enables users to easily and quickly locate objects of different data types, such as files, printers, directories, mail folders, or libraries of clip art. It includes a query function so users can search based on document content or properties (for example, a keyword, the author's name, or a particular phrase).

Task Assistants. This set of components allows the end user to graphically program a series of actions, and then modify them and play them back. Moreover, users can define procedures which are executed automatically in response to a specific event (such as sorting mail into appropriate folders).

User assistance. The UI's new information system enables users to quickly access context-sensitive, relevant information about a task or process they are trying to complete.

Distributed services and storage systems

Object File System. The Object File System (OFS) is a new high-performance file system for Cairo; it offers many features suited for native storage of objects. The OFS stores object data efficiently since it stores small objects with little overhead. It supports multiple streams that allows efficient use for OLE embedded objects. The preferred choice of Cairo's multiple installable file systems, the OFS indexes file contents and properties to support queries. It also supports long file names, quotas and a log of operations to maintain file system integrity.

Access Control. To properly control access to files in the OFS, the distributed security system provides the local operating system with the security identity and access privileges of networked users. Under the distributed security system, a user logs onto the system only once during a Windows Cairo session to establish access to enterprise resources.

Content Indexing. By indexing files according to their content and properties, the OFS provides very fast retrieval of files based on these characteristics. Coupled with the Explorer, content- and property-based retrieval greatly eases finding (organization and presentation) of unstructured information, especially on a network.

Replication. Cairo replicates files in the Object File system to increase availability, and uses the same system to replicate network domain controllers.

Distributed File System. Built on top of the OFS, the Distributed File System (DFS) maps all shared disk space in a network in the form of a single, logically arranged namespace. The DFS combines any number of networked file systems (including heterogeneous file systems) and object stores into a single global namespace. This enables users to access local or remote objects -- such as files, documents, printers, and other users -- using a single naming scheme.

Directory Service. Cairo leverages the DFS to provide a Directory Service (DS) data base describing the various resources on the network. The DS does away with the other naming schemes such as drive letters, E-Mail names, disk directories, file shares, printers, or network domains (C:, John Doe@Host, \WINDOWS\SYSTEM, \MTK\PUBLIC\, LPT2:, or "Building 5"). The DS enables administrators to locate, monitor, and manage all information on the entire networked system, including foreign networks.

Object management (CairOLE)

Cairo will have integrated support for an object oriented file system, programming model, and user interface. Cairo uses an object oriented UI and programming model that is *compatible with*, and an evolutionary step forward from the OLE 2.0 component object model.

With CairOLE, shell-level objects are actual OLE2 objects. CairOLE also supports extensible interfaces between the client and server objects. Most importantly, CairOLE allows the client and server objects to be executed as different processes, including distributed processes (running on different Cairo machines) while preserving all security privileges and restrictions.

System management

Cairo will provide a system management infrastructure to assure effective administration of networked applications (including operating system updates); easy configuration of Windows applications for shared use on a network; inventory and asset management; remote configuration; and extensibility to permit integration of future tools. As it is most applicable to the MIS system administrator, system management functionality is in the Cairo Advanced Server package, and not in the Cairo Advanced Desktop package. Additional tools and utilities are being planned for the resource kits targeted at system management professionals.

Software development support

Cairo Design Environment. The Cairo Design Environment (formerly known as Delphi) enables system administrators and MIS programmers to add behavior and properties to folders – hence the term "smart folders". The CDE also allows these designers to customize the presentation views presented in the Explorer.

Object Composition. Component-based software development allows users to build and compose "pluggable" object components to make applications.

Construction Site. The visually oriented Construction Site forms editor enables developers to combine object programming components and package them as components again for reuse as building blocks.

International language support

Cairo is architected using Unicode as the internal text representation and is being architected to enable easy localization of all system components. Localized versions include:

- *Simultaneous shipment:* The English, French, German, Japanese, Swedish, Dutch, Spanish and Italian versions of both *Cairo Desktop Package* and *Cairo Server Package* ship simultaneously.
- The Norwegian, Danish, Finnish, Portuguese, Chinese (Taiwan), Chinese (PRC) and Korean versions of Cairo will be shipped within two to six months from the first Cairo release.

Cairo Applications

These attributes define a Cairo application.

Minimal Cairo Application. A basic Cairo application uses Win32™, OLE 2, and other existing WOSA features such as MAPI, ODBC, and RPC. It is important to note that existing applications do not have to be modified to leverage system services such as content-indexing (Cairo will supply filters for popular document types), replication, and the ability to have their documents in smart-folders.

True Cairo Application. These applications are built out of components, with extensible interfaces so that the components can be application building blocks. The applications can be enabled to be distributed objects with remote invocation. A Cairo application supports the new Cairo interfaces, such as content filters, smart containers, good shell integration, event notification, query interfaces, and the new help system.

Schedules

These are calendar, rather than fiscal, dates.

Item	Date	Note
PDK-1	Q4, CY93	
Beta (Includes PDK-2)	Q1, CY94	New betas every quarter
Final Product, Release to Market	October 18, 1994	

Packaging

Retail packages for Windows Cairo

Cairo Advanced Desktop. A easy-to-use advanced desktop operating system that features all client capabilities (including file and print sharing), but will not provide domain or enterprise-wide network services. It is designed to be the complete client for Cairo servers.

The target market includes all end-users and developers.

Cairo Advanced Server. A true superset of the Advanced Desktop package this is a system that supports domain and enterprise-wide network services for client-server applications, security, directory services, distributed file systems, resource sharing, and all system management tools and services.

Target for the Advanced Server package are all Cairo servers in the large enterprise and those who administer these servers.

Cairo Design Environment: This package provides the design environment for authoring smart-folders. It includes the core development tools including debugger, form design tools, and all documentation on the environment and Visual Basic for Applications.

Open Packaging Issues

Presentation Manager support: Add-on kit providing support for 16-bit PM applications.

Windows Cairo Resource Kit(s)?. This is an administrative supplement that includes such utilities as performance monitors and workstation account management, as well as selected documentation. An issue remains as to whether there is more than one of these kits -- should there be a resource kit specifically aimed at the CDE designer -- similar to the Lotus Notes 50 sample apps kit?

Pre-release kits and packages

SDK	Software Developer's Kit
DDK	Driver Developer's Kit
Beta for Desktop	To become a retail package
Beta for Server	To become a retail package
OEM Kit	For Windows Ready-To-Run™ and Hardware Abstraction Layer.
HW Compatibility Test Suite	For OEMs
Demo CD	For Microsoft Sales

Hardware Requirements

The ***minimum*** end-user configuration for Intel based platforms:

Item	Desktop	Server
CPU/Speed	Intel 386/25mhz	Intel 486/33Mhz
RAM	12 Mbytes	16 Mbytes
Hard disk	100 MBytes	300 MBytes

The ***recommended*** end-user configuration for Intel based platforms:

Item	Desktop	Server
CPU/Speed	Intel 486/Pentium/Risc	Intel 486/Pentium/Risc
RAM	16 Mbytes	32 Mbytes
Hard disk	200 MBytes	300 MBytes

Processor Platforms

Windows Cairo runs on the same platforms as Windows NT. These are:

- *Intel* Intel architecture systems. (386, 486, Pentium™ series) and compatible processors.
- MIPS Silicon Graphics Inc's MIPS architecture systems. (R4000 & R4400 series processors).
- Alpha AXP Digital Equipment Corporation (DEC) Alpha AXP series.
- Clipper Intergraph Corporation's Clipper series.

Compatibility

Windows Cairo is compatible with applications written for these platforms: MS-DOS, Windows-16 API, Win32, OS/2 1.0 16-bit character mode, OS/2 1.0 16-bit Presentation Manager, and IEEE POSIX spec. 1003.1a interfaces (same as those in Windows NT 3.1).

Features not Considered for Cairo

These non-goals determine the Cairo boundaries:

- a new system software foundation that is not compatible with Windows.
- a user interface based on a flat storage model (equal-instance, sea of files model).
- multiple interactive users.
- using a Cairo-specific hardware platform.
- compatibility with OMG CORBA.
- compatibility with UNIX through the POSIX system. (System VR4, OSF/1, POSIX).

MS 004291

II. Requirements: General

Windows NT Enhancements - Daniel Conde, Bob Muglia

Cairo is the second release of Windows NT. As such, there are a number of features which were not implemented for the initial release of Windows NT which will be completed for Cairo. In addition, Cairo needs to stay in-sync with new releases of MS-DOS and MS-DOS based Windows (Chicago).

Goals

Make the Cairo system behave consistently to the users who are accustomed to other Microsoft Windows and MS-DOS products. This includes commands users type in a MS-DOS Window, peripheral equipment set-up, etc.

Benchmark

Microsoft's previous Windows products set the expectations on how compatible our new systems software and releases are.

Features

Chicago Enhancements

- *Long Filenames on FAT.* Chicago will enhance the FAT filesystem to support long filenames. Cairo will fully support these enhancements in our FAT filesystem.
- *Command environment.* Cairo will provide the Chicago command.com enhancements:
    - *File Search engine* for various commands with different wild card specifications, content based searches and attribute based filters.
    - Enhancements to the *copy* and *dir* commands. The *dir* enhancements are new information displays. Copy enhancements are minor.
    - *Toolbar and font enhancements.* Cairo will support the command toolbar and scalable font support enabled in Chicago.
- *Control Panel.* Cairo will provide at least the same set of control panel objects as Chicago. This includes additions to Windows 3.1 such as Handicapped Access, Power Management, Modem and Phone controls.
- *Plug and Play.* Track all hardware setup and configuration activities to make setting up and configuring hardware easy.
- *Display support.* Track the new desktop/display panel functionality, frame grabber, dynamic screen dimension switch.
- *Printing support.* Track bi-directional printer support, and integrate support for Windows Printer Group projects.
- *Layered Comm Driver.* Implement the equivalent of the Chicago layered comm driver in Windows NT.
- *Device independent color.*
- *Tiny font in DOS boxes.* (Wish list in Chicago) If Chicago includes this, we need to integrate our globalization system to this.

ASTRO (MS-DOS 6) compatibility

Chicago will integrate Astro changes, and Cairo need to add these as well. The MS-DOS 6 disk compression feature is planned for Windows NT 1.1, thus it is covered below.

- *Choice (Ync).* A small batch file command for getting user confirmation.
- *Interlnk/Intersvr.* A utility used mainly for Portable to Desktop file transfer.
- *Floppy BackUp.*
- *Anti-Virus.*
- *Power Management (APM 1.0)*
- *Enhancements to existing MS-DOS commands.* Change DOS Key, add thousands place-separator, paged output switch,
- *PSS Supportability.* Provide the same level of troubleshooting tools. We plan to have a Cairo Product Development Consultant from PSS to help provide requirements.

Pen

Windows for Pen Computing Extensions. Include the Pen extensions in Cairo.

WOSA APIs

- *ODBC*
- *MAPI 1.0*

Windows NT 1.1

The following features are committed or under consideration for Windows NT 1.1. If they are not completed for NT 1.1, they will be included in Cairo.

- *Netware clients.*
- *Disk Compression.* Full support for MS-DOS 6 compressed volumes.
- *32-bit OLE 2 support* with full 16/32-bit interoperability. In Cairo, we will use the distributed binder infrastructure to support OLE 2 apps.
- *Japanese Product* (actually NT 3.1J) - DOS V, Japanese IME, combined font for user defined characters, DBCS and Unicode enabling.
- *Paging the kernel and kernel mode drivers.* Make key drivers pageable.
- *Access pack* Support for handicapped users (sticky keys, etc.).
- *Plotter Driver.* Support all plotter characterization files from Chicago.
- *Automatic IP Address Configuration.* This capability will be integrated into the Cairo directory service.

Windows NT Product 2

The following features are considered "Product 2 Features" by the NT group, as such they should be included in Cairo.

- *Partitioning on the fly.* Support re-partitioning without a reboot. Grow volume sets dynamically on OFS.
- *Post Mortem Debugger for kernel.* Capture the exception and write data to improve supportability.
- *General dump utility for apps.* General exception handler and core dump analyzer to diagnose application errors after they occur.
- *Diagnostic utilities for PSS.* System support for more error logging information (similar to the Windows 3.1 Dr. Watson) to aid PSS in the debugging of problems.
- *Support for mapped Istream.* Provide support to allow the OLE Istorage/Istreams to utilize the Win32 memory mapped file architecture.
- *Attributed kernel profiler.*
- *New Internet daemons.* Telnet, FTP, plus more stuff.

- *RIPL Server* for MS-DOS workstations.
- *Unicode CMD.EXE.*
- *3D Graphics.* Integrate OpenGL into Win32.
- *More robust shutdown.* Flush the registry buffers during shutdown.
- *Multiple monitor support.* This enables an application debugger to run on a second monitor.
- *Mouse Trails.*
- *High-end pointer support.* Tablets, for example.
- *GDI Backing store.* Allow an app to draw off-screen.
- *Shared Tape Devices.* Currently, only files and printer devices can be shared. Supporting shared tape drives would dramatically increase the capacity of a backup server.
- *Support for remote console applications.* Currently, it is extremely difficult to write an efficient application which supports remote screen and keyboard control of Windows NT workstations. Although we don't plan to ship such an application with Cairo, we will enable 3rd parties to build these apps.
- *Integrate RAS and Mac support into the Cairo product.* RAS and MAC are planned for shipment in the Windows NT Advanced Server. RAS support will be directly integrated into both the Cairo desktop and Server. Mac support will be an integral part of the server.

Additional features outside the Cairo team

These features should be completed in the Cairo timeframe but will be developed by groups outside the core Cairo team:

- *New transports.* OSI and DECNet
- *Unix or additional POSIX support*

Benefits

Cairo represents the second major release of Windows NT. As such, there are a number of items which customers expect in an operating system which were not completed in the Windows NT 3.1 release. Providing support for these missing features is an important customer requirement.

Non Goals

- *Remote IPL of Cairo workstations*
- *Generic hardware support for DOS/WOW.*
- *Multiple simultaneous interactive users*
- *Support for Pen APIs*

Under Consideration

- *Comm port sharing*
- *Mobile Windows support.*
- *PCMCIA support.*

Application compatibility - Bob Muglia

Cairo will support MS-DOS, Windows, Win32, 16-bit OS/2, and POSIX applications.

Goals

Cairo will allow users to run the vast majority of their existing applications. In most cases, the applications will run almost as well under Cairo as they did in their native environment. In some cases, Cairo will provide the user with more functionality from their existing applications.

Benchmark

The benchmark for application compatibility is always the native environment. Thus the benchmark for 16-bit Windows applications is Windows 3.1, Win32 apps is Windows NT 3.1, 16-bit OS/2 apps is OS/2 1.3.

16-bit Windows application will perform no worse then 10% slower under Cairo. 16-bit OS/2 applications will perform no worse then 20% slower under Cairo. Win32 applications will see a performance increase when compared to Windows NT 3.1 due to improvements made in the underlying OS. Chicago applications will perform as well on Cairo as they do on Chicago.

Compatibility features

- *16-bit Windows applications.* 16-bit Windows applications will run on Cairo without any special user configuration. They will only be able to drag-and-drop onto the Cairo desktop if they are OLE 2 enabled.
- *Win32 applications* All Win32 applications which run on Windows NT 3.1 will run on Cairo. They will only be able to drag-and-drop onto the Cairo desktop if they are OLE 2 enabled.
- *Chicago applications.* A Chicago application is a Win32 application with OLE 2 support. Additionally, the application may take advantage of MAPI 1.0, ODBC, RPC, or Chicago features like device independent color. Chicago applications will run on Cairo with drag-and-drop supported to the Cairo desktop.
- *MAPI, ODBC.* Applications with utilize MAPI or ODBC are supported on Cairo.
- *OLE 2.* Both 16-bit and 32-bit OLE 2 applications are supported on Cairo. In addition, full interoperability between 16 and 32-bit OLE 2 applications is supported.
- *MS-DOS applications* Both character and graphical MS-DOS applications will run on Cairo. This includes support for VGA graphics, both full-screen and in a Window.
- *16-bit OS/2 character applications.* 16-bit OS/2 VIO applications are supported on Cairo.
- *16-bit Presentation Manager applications.* Although not included with the base product, Cairo will support 16-bit OS/2 Presentation Manager applications with the addition of the PM subsystem. Presentation Manager applications run in a separate desktop providing only clipboard interoperability with Windows applications.
- *POSIX applications* Cairo will support the same POSIX applications that Windows NT 3.1 supports. Although POSIX applications can run on any file system supported by Cairo, the only file system which supports POSIX semantics is NTFS..

Benefits

Compatibility with Windows and Win32 applications ensure that the largest base of GUI applications run on Cairo, thus allowing the user to easily move to Cairo. MS-DOS application support means the user doesn't have to give up the countless MS-DOS based applications and utilities still in use.

16-bit OS/2 application support allows users with investments in OS/2 to move to Cairo and also provides developers with support for OS/2 development tools.

Non goal

Cairo will not support any MS-DOS, 16-bit Windows, or OS/2 application which directly accesses the hard disk or non-standard devices. This does not include devices for which Windows NT provides a virtual interface, such as a VGA, serial or parallel ports. In addition, like Windows NT 3.1, it is possible to support these applications if the vendor has written an NT device driver and exposed the hardware through a virtual device driver (VDD). For example a 16-bit Windows 3270, FAX or scanner application can be supported through a VDD.

Cairo may not support 16-bit Windows applications which access internal Windows data structures directly without calling a Windows API. Also, applications which call internal, undocumented Windows APIs may not be supported.

Applications which call the NT API directly (bypassing Win32) may not be supported as these interfaces may change with Cairo.

32-bit OS/2 applications are not supported.

National Language Support - Giovanni Mezgec

The term National Language Support (NLS) refers to hardware and software provision that make personal computers accessible to users in different languages and countries. The basic objective of National Language Support in Cairo is to leverage the existing localization work done in Windows NT and improve it as appropriate.

Goals:

- Deliver high quality French, German, English, Japanese, Swedish, Dutch, Spanish, and Italian versions of Windows Cairo simultaneously.
- Increase penetration in international markets by establishing name recognition and a reputation for high quality international versions.
- Provide ISVs with a well designed methodology and tools to localize their Cairo applications.

Benchmarks:

Advanced NLS support: Novell - Ease of internationalization: NeXTStep - Scripts, keyboard support, and timely releases of international versions: Apple

Project wide issues:

- *Simultaneous shipment:* The English, French, German, Japanese, Swedish, Dutch, Spanish and Italian versions of both *Cairo Desktop Package* and *Cairo Server Package* will ship simultaneously. This means RTM (Release To Manufacturing) for all these versions will take place within 0 to 3 weeks after US release. The Norwegian, Danish, Finnish, Portuguese, Chinese (Taiwan), Chinese (PRC) and Korean versions will be shipped within two to six months from the first Cairo release.
- *No-compile localization and localization tools:* Cairo will support no-compile localization for all resource data and property-name data. No-compile means that any localizable strings, messages, dialog boxes, etc. must be easily accessible and isolated from code. The Windows resource model offers the best environment for an easy identification, access, and manipulation of resources. In terms of tools, Cairo will use an improved version of RLTools for its localization effort.
- *Other Localization Tools:* Because Cairo introduces several evolutionary technologies in several areas such as help files, CBTs etc, most of the current localization tools developed for previous technology will be no longer useful. Cairo must provide appropriate new tools to replace those that are obsolete. This becomes even more strategically important because of Advanced Systems' general strategy to have most of the localization done outside Microsoft by external vendors. Vendors are going to need appropriate tools to perform localization tasks in a timely matter and according to Microsoft quality standards.
- *Storing language information:* Cairo encourages users and applications to add properties for NLS data on their objects. The advantage is to provide a unique way of identifying the language of an object, information which is used when language dependent operations are performed. Clear examples for possible use of this can be found in the stemming and spelling checker mechanisms.

- *System properties:* Cairo will use the Unicode character set to display character data for property names. Cairo will also provide localized property names stored in a TypeLib.
- *Fonts*: Windows NT 1.0 J development plans include the design of an End User Defined Character (EUDC) support for the Japanese version. Possibly this architecture will be extended to include more general font *linking*. Cairo will leverage the Win NT architecture to build a more sophisticated font mechanism, providing a way to let users and applications know the character content of a font.
- *Edit Control*: Cairo will contain a Unicode aware Rich Text Object Edit Control. It will also support Japanese scripts (running from the top to the bottom). This functionality will be included in the generic code base only if performance is unaffected.
- *Keyboard*: Cairo will leverage the Win NT and Chicago Keyboard architecture. It will provide a mechanism to quickly switch between multiple keyboard layouts by improving the dual mode layout supported by current Easter European versions of Windows 3.1.
- *Common system services*: Examples of these are the File Open/Save dialog boxes, and default property-sheet. Typically they appear to the user as belonging to the application from which they are called, with the result that the application in use may then appear to have a mix of presentation languages. Cairo will solve these UI problems.

Cairo National Language Support by Component

National Language Support issues are spread to all Cairo components. This section describes the main NLS issues related to some of the major Cairo components. It outlines major NLS guidelines, but it is not exhaustive in identifying all NLS specific concerns, since more issues are expected to arise as progress is made in the product development.

- *User Interface:* The Cairo shell will contain a number of user-interface objects eligible for localization. Specifically, the user interface items that have to be localized are: property names, context menus, conventional menus, string tables, accelerator tables, dialog box templates, icons, cursors, and bitmaps. All Cairo shell components will display\accept text strings via Unicode characters. All Cairo objects will be easily localized and designed to meet usability needs of international users. The user interface will help users switch and select locale information quickly and efficiently, at least within the boundaries of Latin-1 based languages.
- *Cairo Design Environment:* The Cairo Design Environment will permit authoring of collections in the local language, including scripting of behavior. Collections authored in one language will be usable from any language client. Views will display appropriate date/time, currency and numeric formatting for the locale of the client. Provisions for locale-specific forms, property names, and views will be also made. Data stored by CDE generated form fields (custom controls excepted) will all be stored in Unicode, or in appropriate location-independent format.
- *Object File System* The OFS Kernel exposes the object store through the OFS API. Some of the functions in the API manipulate object names, volume names, paths, etc., all of which will be represented with Unicode text strings.
- *Content index:* This component has many NLS implications since there will be a method for indexing and retrieving documents in any language and multilingual documents. Besides the awareness of locale and language information, the content index has major components that are language dependent: word breaker, word stemmer, and word normalizer. Cairo will supply Word Breaker, Word Normalizer and Word Stemmer functionalities for major languages depending on business case; or encourage independent vendors to supply additional modules for languages not covered in the initial plan. Word Breaker, Word Normalizer and Word Stemmer CairOle interfaces will be made publicly available, so that applications can be written to take advantage of their services.

- *Distributed File System and Directory Service:* This s by far the Cairo component most affected by NLS issues. In a large namespace, we can easily assume the presence of different languages and locale data settings and possibly non-Unicode aware objects.
  For example; 4 users are running different language versions of Cairo server: US for User A, French for User B, Swedish for User C and Japanese for user D, and we want to perform the following operations:
  1. D wants to log into B: Is it possible?
  2. A looks at a directory of files in B: What sorting order is used? The user would expect to see English sorting order.
  3. A looks at a summary catalog generated on A: The user would expect an English sorting order, but the French server is the one generating the catalog.
  4. A does a query on the namespace, assuming to use the English stemmer.
  5. A volume is replicated between B and C: The indexes are rebuild on each server, meaning that B presents one sorting order, C presents a Swedish sorting order.
  6. An error messages is sent from C to D: In which language this is going to be displayed? D would expect to see Japanese.

  At this development stage, there is not a unique and solid solution to these examples. In general, DFS will understand the different language settings and react to them efficiently and as transparent as possible to different users.
- *Mail systems:* The whole Cairo mail system will be Unicode aware and Unicode characters will be sent between Cairo mail clients and servers. The Cairo mail server will perform the necessary conversions if the target destination is a non Cairo system running on a different code page.
- *System Management:* System management utilities will be Unicode aware.
- *Development support:* All Windows Cairo SDK tools will be Unicode enabled as is the C compiler.
  To help ISVs localize Cairo applications, the SDK will have a basic set of localization tools:
  1. No-compile localization tool for all resource data;
  2. International Software Development Handbook;
  3. Subset of the Cairo Glossary for the sim-ship languages which contain the main translations for Cairo UI components.

Under consideration

- *Conversion Utilities for Cairo Text Editor:* Using the Win 32 NLS APIs and appropriate conversion tables, Cairo Text Editor will have a mechanism to convert different file code pages. For example, the Text Editor can very simply leverage an already existing mechanism included in the Win NT SDK to convert to\from Unicode.
- Anything else that supports international ISVs and end-users but requires minimal use of development resources. Work with subsidiaries, international PSS and international ISVs to identify issues.

Non goals

- Cairo will not provide Language Packs.

Packaged Product Documentation- Paul Goode

Cairo packaged product documentation will address the needs of two groups of users: end users and system administrators. Documentation for end users will focus on helping them find, produce, edit, organize, and disseminate information; documentation for system administrators will focus on helping them plan, install, and maintain Cairo-based networks, as well as support the needs of end users connected to those networks. For detailed

information regarding Cairo packaged product documentation objectives, strategies, and components, see the *Cairo Documentation Plan*.

> *Note:* Documentation requirements for the Cairo Design Environment are unknown and are being assessed.

Goals

Packaged product documentation focuses on end users and system administrators. The specifics of packaging have not yet been determined, so it's possible that some of the titles below may change. For the administrator documentation in particular, it may be more useful to think of the titles as key audience needs that must be addressed, regardless of how they are eventually packaged or combined.

The goals of Cairo packaged product documentation are to

- Get users up and running quickly
- Bring the benefits of Information at Your Fingertips home to users
- Make Cairo documentation a usable, enticing tool for user assistance
- Support systems administrators working in a variety of environments

The Cairo documentation set includes both online and paper components. Paper components include manuals for both end users and system administrators Online components include object help, diagnostic and troubleshooting tools, task wizards, and online tours.

End User Documentation

The paper component of the end-user documentation set will emphasize broader tasks that require extended discussion or span multiple procedures (for example, formulating a query) and conceptual information for users who wish to be fully productive in the Cairo environment. It will include these components:

- ***Quick Start Pocket Reference.*** For all users who, regardless of level of experience with computers in general or Windows in particular, want to get up and running quickly. This reference will feature concise procedures without conceptual information. *Estimated length:* 6-8 panels.
- ***User's Guide.*** Steps users through *essential* Cairo procedures. This book is a step-by-step tutorial aimed at users who, whatever their experience, want to take a more leisurely approach to learning Cairo or who require more hand-holding in the process. Conceptual information will be limited to the minimum necessary to step users through the various procedures. *Estimated length:* less than 100 pages.
- ***User's Reference.*** Documents *all* Cairo procedures and provides conceptual information -- in other words, a Cairo "User's Bible." Topics will be organized according to major task groups and within each task group by module; for example, Configuring Your System, Customizing the Desktop, Sending Messages, and so on. *Estimated length:* 250-300 pages.

The documentation's online component will provide users with on-demand detailed assistance in carrying out specific tasks -- for example, changing settings in a property sheet or moving a document to the desktop. Online documentation will also include introductory tours and task wizards to assist users in learning or completing selected complex tasks. Cairo online documentation for end-users includes these components:

- ***QuickInfo.*** Provides quick information about objects and user interface elements; in other words, users can get information about whatever they can point to. It will contain concise, context-sensitive information about objects on the Desktop, such as files and folders, and user interface elements, such as title bars, scroll bars, dialog box fields, and menu and button commands.
- ***Cairo Help.*** Contains general procedures that may not be tied to a specific dialog box or property sheet, more detailed information about a particular field in a dialog box or property sheet, conceptual information, and command reference topics for NT and Object Basic language elements.
- ***Error Messages and Troubleshooting.*** Includes information such as what happened, why it happened, and how the user can recover from the error.

- ***Tours.*** Two modules will introduce Cairo and provide mouse training to both novice users and experienced users who are moving to Windows. The mouse tour will introduce the basic mouse skills needed to operate the Cairo interface. This tour will lead into a second unit that will orient users on how to get work done in Cairo, covering concepts and features essential to getting up and running *(e.g.*, the Desktop, the Explorer, Cairo Help, and the transfer mechanism). A separate tour aimed at Windows upgrade users will help transfer their previous Windows experience so that they can be up and working with minimal disruption. It will also introduce new Cairo features.
- ***TaskWizards.***[2] A Cairo TaskWizard structures a task a user wants to accomplish and automates the task in a way that removes the burden of needing to know the process. For the user, a task usually entails more information than a specific Cairo command or behavior it is simply something the user wants to accomplish. Because of resource constraints and uncertainty of tools, we will provide a small, strategic set that will provide high return to the user and reinforce the notion that Cairo is easy to use.

  Advanced Systems User Education proposes the following Cairo TaskWizards for Cairo first release: (1) a custom Desktop wizard to guide users through the options available to customize their desktops, including fonts, wallpaper, color schemes, sounds, the results to be saved as specific sets of desktop properties; (2) a printing wizard to display a list of printing tasks, including configuration and option setting based on the specific printer(s) available to a user. Users can then deal with more clearly-delineated tasks one-by-one; and (3) a search wizard to structure the searching task as a set of sequential choices and to provide enough information for decision guidance.

System Administrator Documentation

The online and paper documentation for the system administrator work together in the same way as the end-user documentation. Online and paper documentation both will provide complete information for those tasks that are essential for system management, thereby sparing administrators from having to rely exclusively on either for critical information. For optional or advanced tasks, administrators will consult paper documentation for conceptual and high-level procedural information, and online documentation for detailed instructions on completing dialog boxes.

Paper documentation for network administrators includes the following components:

- ***Administrator's Roadmap.*** Shows administrators how and where to get different types of information from the administrator documentation set.
- ***Simple-Network System Administrator's Guide.*** A short, easy-to-use manual geared towards novice administrators who are responsible for simple client-server networks. The manual will enable administrators to quickly get their networks and their users set up and running.
- ***System Administrator's Reference.*** Explains how to perform the administrative tasks of configuring and managing a Cairo system. The manual will contain only the conceptual and procedural information necessary to complete the tasks.
- ***Migration Guide.*** Assists administrators migrating to Cairo from other network operating systems. It will also cover migration from peer-to-peer networks or adding peer workstations as clients.
- ***Planning and Installation Guide.*** Helps administrators plan and install Cairo in network environments. The manual will also contain information about network device drivers, protocols, and network cards.
- ***Comprehensive Glossary and Index.*** Provides an alphabetical listing and definition of Cairo terms, as well as a comprehensive index to the rest of the administrator documentation. An administrator will be able to look up "security," for example, and find brief accounts about different types of security and how each type works. The reader will also be referred to the appropriate pages in the planning and administrative guides for further information.

Online documentation for network administrators includes the following components:

[2] Note that Legal has specific guidelines for the use of the term "wizard." The word "wizard" is referred to as lowercase. All Microsoft wizards must be called by a two-part name with the word "wizard" preceded by a descriptive term, such as PrintWizard, SearchWizard etc. For now, we'll refer to them as Cairo TaskWizards.

- ***QuickInfo.*** Provides specific, context information for system administrators.
- ***Cairo Help.*** Provides topics for the system administrators that will be a superset of what the end-user gets. For example, there will be topics associated with the system administration tools that end-users won't have access to. System administration topics will also be written on a more technical level.
- ***Error Messages/Diagnostic Information.*** Features the same error recovery system as described for end users. Error recovery will be as detailed and as descriptive as possible to meet administrator's needs.

Benefits

Cairo User Education anticipates that this approach will provide these benefits:

- Reduced dependency and length of paper documentation for end users.
- Network documentation consolidated into a thematically coherent set focused on the central administrative issues of planning, installation, maintenance, and troubleshooting.
- Enable fast, productive use of Cairo by end-users and administrators. We are addressing ongoing customer requests for more network planning and troubleshooting information.
- Improved access to immediately relevant information via task reference, task help, and the comprehensive index.
- Reinforce the impression of Cairo as a user-friendly environment.

Non-Goals

We are specifically not planning on an extensive set of CBT lessons. Instead, we are limiting this effort to Cairo orientation and a selected set of basic skills. Also, we are intentionally avoiding a comprehensive user's reference to Cairo features and the UI. We will instead concentrate on the tasks essential to productive work.

Under Consideration

Selected parts of the documentation will be shipped as part of the Cairo Resource Kit. Advanced Systems User Education will evaluate and decide on this during calendar Q293.

Also under consideration are task Wizards system administrators. These will be addressed when the administrator interface is defined. Finally, a number of other possibilities for end-user wizards have been suggested, but require additional investigation. Candidates include file backup, file sharing, collection building, and end-user programming tasks.

III. Requirements: User Interface

User Interface - Steve Madigan

Goals

Ease of Use. The Cairo user interface will be as easy to use as any other system for common desktop-oriented tasks: document production, email, etc.

Ease of Learning. The UI will be easy to learn, and require no learning for the tasks users already do today. New functionality must be easily discoverable.

Consistency. The smallest number of UI paradigms and concepts are applied as broadly as possible, avoiding the need for special purpose interaction methods as much as is possible.

Extensibility. The shell will be extensible at many levels including the types of objects, the commands on objects, the properties on objects, and the views available for looking at collections of objects.

Migration. The UI will be "comfortable" for existing Windows 3.1 users.

Finding Information. The UI will be easy to use for finding information on a workstation or in a networked environment. The focus on finding information is simplicity of model and use, exploiting content and property based retrieval.

Benchmark

Ease of use for common desktop application scenarios and file management is measured against Macintosh system 7. Speed in the user interface in general is measured against Windows 3.1 (DOS and NT versions).

Features

- *Animation*. The shell makes use of various animations as feedback, especially for transfer operations (e.g. move and copy).
- *Applettes*. The Windows 3.1 applettes are provided, with UI improvements to fit with the Cairo UI model. New applettes (TBD) are also included.
- *Application Consistency*. Common operations, paradigms, and commands are consistent with those defined for next generation Microsoft applications (and others who follow our design guidelines).
- *Common Dialogs*. The shell provides common dialogs which support the same level of functionality as the Explorer or other standard system component.

  File.Open (and Insert File)

  File.SaveAs

  File.Print

  File.PrintSetup (to select a printer)
- *Context-Sensitive menus*. Objects have context-sensitive popup menus which present commands on objects based on the type of the object, and its container. Context-sensitive menus are also present on window mini-icons (in the left corner of the title bar) so that operations can be performed on an object when it is open in a window.
- *Control Panel*. The control panel is changed to no longer be "special", instead presenting the contents of the Win3.x control panel as objects with properties and/or folders. All standard control panel items included with Windows 3.1 NT are provided.
- *Desktop and Tray*. The desktop is capable of containing (storing) objects of any type, including links. The desktop also has a "tray", which serves as the home for a system menu and as a convenient location for frequently used objects.
- *Drag and Drop*. The UI makes extensive use of drag and drop to accomplish transfer operations (move, copy, link, print, etc).
- *Explorer*. The Explorer supports Windows 3.1 file manager style browsing over objects, plus content and property based filtering and queries. The standard views included with the Explorer are Icon and Small Icon, List, and Table. The views in the explorer are extensible by the containers that the explorer views, allowing customized views. The filters used by the explorer for querying are extensible.
- *Inspectors*. Inspectors allow the user to view information about the currently selected object(s). Both property (property sheet) and content (preview) inspectors are provided.
- *Information Cursors*. Information cursors place information about objects near the cursor, including help or object-descriptive information, name, or a visual "thumbnail".
- *Links* Links in the shell allow the user to have quick access to an object that is stored elsewhere.
- *Long Names*. Objects can have long names instead of today's 8.3 names.

- *Menus.* Enhancements in menu items and behavior include default menu items, tracking of menu selection even when no mouse button is held down, and improved keyboard behavior.
- *Multi- and Single-Line Edit Control.* The text edit control used is enhanced to support drag and drop ("in" and "out") and a context menu for transfer operations.
- *Network Access.* For Cairo networks, access to all network resources (disks, printers) without the need for establishing "connections" is provided. Downlevel networks may still require making connections.
- *New 3D Visuals.* The system provides a new standard 3D look for windows, standard controls, and borders.
- *Objects.* Objects of all kinds can, in general, exist in any system container. The need for special purpose "managers" (e.g. print manager, address book, program manager, etc) is removed.
- *OLE2 integration.* The shell is compatible with, and integrates with, OLE 2 interfaces and UI paradigms where applicable, including:

  Types, and emulation and conversion of types

  Use of OLE objects as values of properties
- *Printing.* Printers are objects, and can be drag and dropped to. By default, the printers the user utilizes most frequently are kept in a special printers folder, which makes them available through application printer dialogs (e.g. PrintSetup); however, references to printers may be kept anywhere the user chooses for easy access. Applications will support printing to specific printers from the shell, e.g. by drag and drop. Users will be able to find network printers (by browsing or property-based query), and use them without manually establishing connections.
- *Program Groups and Items.* Program Groups and Items in the Windows 3.1 program manager are converted to folders and objects which may exist in any container.
- *Properties and Property Sheets.* Property sheets allow the user to view and edit properties on objects. Properties are divided into sets which are presented on "pages". Standard property pages are supplied on all objects for general storage system properties, owner, and notes properties, and access permissions. Additional property pages for specific system types (e.g. users) are also supplied. The system, ISVs, and users can define new properties and property sheets.
- *Proportional Scroll Bars and Resize Grips.* The standard system scrollbar is changed to use a proportionally sized "thumb", which shows the relative size of the visible display of a scrollable field to the full field contents. An enlarged resize "grip" is provided on windows when there is space in the lower right corner.
- *Recent Folder.* A special "Recent" folder is automatically maintained by the system and contains links to recently used objects.
- *Rename in place.* When the name of an object is presented (e.g. under an icon), the user will be able to edit the name directly, in place.
- *Scraps.* Shell containers (desktop, tray, folders) can contain stored clipboard contents (including OLE objects) called "scraps." This allows transfers between applications via shell containers, as well as reuse of document parts.
- *Send/Send Folder* A Send menu item is available on all objects that can be transferred. The send item is a cascade menu which lists possible destinations that an object can be sent. These destination objects are indicated by the user placing references to them in a special send folder. Send is a shortcut for drag and drop.
- *Sound Integration.* Interactive and environmental sound schemes will be an integral part of user feedback.
- *Summary Catalogs.* The shell provides access to summary catalogs (defined in the section on Directory Service).
- *Templates.* Templates are used to create new instances of objects from existing instances.
- *Transfer.* Transfer operations are supported via drag and drop (default and non-default) and commands.
- *Views.* Standard explorer views are provided by the system: large icon, small icon, list, table. Categorization by property values is supported. Views are also extensible, allowing ISV-provided views of standard containers and specialized views of specialized containers.

- *View State Persistence.* View state of objects is maintained persistently on a per desktop basis and in links, so that users see the same view of an object that they last saw, even in a shared (network) environment.
- *Wastebasket* A standard desktop drop target for removing objects that allows the removal to be undone.
- *Windows and Minimized Windows* The appearance of minimized windows is changed to distinguish them from objects.
- *Workspaces* MDI workspaces allow the user to create "projects" of related objects which, when open, have their windows constrained by a master frame, similar to MDI applications in Windows 3.x.

Compatibility

- For application compatibility, see the section titled "Application Compatibility".
- Program Groups and Items will be converted.
- Existing Control Panel items will be supported.
- The Program Manager and Control Panel DDE interfaces are supported so that existing application installers that rely on them continue to work.
- Network connection dialogs provided through WINNET32 are supported for downlevel and foreign networks.
- The Cairo shell will be capable of presenting ISV-defined property sheets and context menus defined and implemented for Chicago. Chicago explorer extensions (e.g. for mail, control panel, etc) will *not* be supported.

Benefits

- The desktop environment is much easier to work with and more powerful than Windows 3.1, which provides no desktop.
- Absolute consistency in the way the same operations are discovered and performed on different object types will make the system much easier to use
- Access to the network is simplified dramatically across the board.
- Finding information on a workstation and on the network is much easier through the provision of integrated browsing, querying, and extensible viewing capabilities
- A single system UI model for properties will make access based on properties much easier and more consistent.
- Users will not need to learn the different user interfaces of the various "manager" applications as they do now
- Consistent transfer operations between applications and filing systems (the shell) will make transfers easier and more powerful.
- The combination of templates and direct manipulation of objects removes the need for the user to ever work directly with applications ("programs").

Non-Goals

- Anything else

Under Consideration

- Bundled file viewers, integrated into Explorer
- New applettes that exploit the new UI
- Desktop organizers (e.g. fileracks) and arrangement schemes that allow for smart, customized object mangement on the desktop
- Live desktop embeddings for clocks, stock tickers, useless but entertaining animation etc.

- Integrated "source-list" navigation support, providing access to the list from which an object was opened off of it's window menu (e.g. the way next/previous work in mail, available generally)
- "Projects", which provide group window operations on a set of objects that the user works on as a group

User Assistance - Valerie Horvath

Cairo User Assistance provides a way for users to get information that will help them in completing tasks and in understanding the operating system or an application. Cairo User Assistance extends the current capabilities of context sensitive help and help article viewers to allow users to get finer grained task-oriented help, and to integrate all of the users help information. In addition, Cairo User Assistance will incorporate wizards and online Tours as standard and easily accessible components of the overall help information. Cairo User Assistance utilizes different methods of providing help to address different user's help information needs.

Goals

The goal for Cairo User Assistance is to provide users with easy, timely, and relevant information about the task in which they are engaged. To facilitate this goal, there are different types of help information available to the user: very quick and specific information (Quick Info), more detailed information (Cairo Help), skill and orientation information (Tours), and interactive task assistance (Wizards). Users are able to access all of this information from the Help menu on the Tray, which is the place users can always go to get Help. There will be other ways to more quickly access these forms of Help, in addition to the Help menu on the Tray.

Benchmark

Competing operating systems provide quick help (Balloon Help on the Mac) and applications themselves are beginning to build in quick help functionality (e.g. Quicken or MacInTax). These provide the benchmark in terms of functionality for Quick Info. The benchmark for Cairo Help functionality is based on today's WinHelp.

Features

- *Quick Info*. Quick Info provides context-sensitive help for any object. This includes objects on the desktop, elements within a dialog box, menu commands, toolbar buttons, etc. When a user accesses Quick Info, a window pops up near the object and an arrow points from the Quick Info window to the object. The Quick Info window contains information that answers "What is it?" "What does it do?" and provides a tip on how to use the object. For example, when a user gets Quick Info on the Print button in the Toolbar, the Quick Info states that the object is a Print button, explains what the Print button does, and gives the user a hint on using the Print button.

  Two methods of using Quick Info are supported: Point-and-Click (help on whatever you point at) and Riffing (users enter a mode where whatever they move their cursor over pops up help; this mode is useful for cautious and new users). Each Quick Info window has a More button which links the user from Quick Info to more detailed information contained in Cairo Help.

  An authoring mode is built into Cairo to allow authors to easily write Quick Info text. The authoring mode displays the Quick Info window without any text. The author then fills in the fields in the Quick Info window and closes the Quick Info window to save the changes.

- *Cairo Help* (Help topic browser and viewer).

  From Cairo Help, users can browse through help files, search for specific help articles, and view individual help articles. Cairo Help extends the functionality present in WinHelp today. Like WinHelp, it displays articles and supports navigational links. In addition, Cairo Help provides access to *all* the user's help files from one location. This means a user can see a list of all help files available to her (Cairo and downlevel) and choose which file is of interest. Note that some of the user's help files may exist on a network and some may exist on her disk. Cairo Help also extends the user's ability to search for help information and to see the organizational relationship between articles (as in a Table of Contents). The main features of Cairo Help fall into two categories: Support for existing WinHelp files and Features which address user feedback.

Support for Existing WinHelp files. Cairo Help will of course run existing compiled WinHelp files. To do this, Cairo will ship the appropriate version of WinHelp, most likely WinHelp 4.0 which ships with Chicago. In addition, Cairo will provide tools which will allow authors to convert the WinHelp RTF source to Cairo Help forms. No new functionality will be added to downlevel Help files. Rather, Cairo Help will focus on providing better access to the information that is contained within downlevel files by providing automatic full-content searching, as well as keyword searching.

Features which Address User Feedback. The existing capabilities of WinHelp will be available in Cairo Help. However, Cairo Help will also offer new functionality. In Cairo Help, users will be able to have multiple windows containing Help articles displayed at a time. This will allow users to refer to multiple articles simultaneously. A common request of users is to be able to see the organization of the Help articles within a file. Cairo Help will use "smart folders" to allow users to browse the organization of a Help file in a separate window than the contents of a Help article are displayed in. The "smart folders" presents the user with a hierarchy similar to a combined Table of Contents and Index which a user can skim to find the interesting categories.

Another common request is to have better access to Help information. In Cairo, users can perform full-content searches over Cairo Help files, or search based on the properties of a Help article (such as keyword) by using Cairo's built-in content indexing and Boolean query capabilities. The content-based searching functionality is available for Cairo and downlevel help files. This capability provides users with better access to information, since they need not rely only on authored keywords which may not be the same as the users' words.

Users will be able to directly copy information in a help file. In addition, users will be able to select, copy, and print more than one help topic at a time. Finally, authors will be able to include OLE objects in the help topic.

Cairo Help will use the Cairo Design Environment to author Help topics. This design environment is form-based, and the Help form would have fields for Title, Body, Keywords, etc. To accommodate internal User Ed groups we are investigating what tools can be provided to take WinWord RTF and convert it to a Help form

- *Tours.* Cairo Tours will be designed to get every Cairo user productive almost immediately so that there is very little "work stoppage" because a user has upgraded to Cairo. Once a user has learned the fundamental concepts and skills, Tours show users how Cairo features can make them even more productive. The approach will be "layered" so that all users, from experts to novices, can benefit from the information provided. Two Introductory Cairo Tours will teach: the minimum "mechanical" knowledge needed to work in the Cairo environment, such as how mouse actions affect objects and the Cairo desktop metaphor, and the 5 to 6 core Cairo skills/concepts such as searching and using Cairo Help. The Cairo Feature Tour will motivate users to investigate Cairo's more advanced features by emphasizing the benefits and ease of use of Cairo. Tours will provide links to the corresponding Cairo Help topic, allowing users the option of getting instructions so that they can use the feature immediately.

  Access to Tours will be from the Help menu and the Cairo Help folder. Specific Tour modules can be searched for within the Help folder. In addition, some Cairo Help articles will include links to related Tour modules, for example a demonstration of how to create a folder..
- *Wizards.* Wizards allow users to perform "expertly" immediately by performing a task for the user. The tasks range from simple automation much like a batch file to guiding the user through a multi-step process for a more complex task. The complexity of wizard is determined by the type of task being addressed. For example, the Custom Desktop wizard will guide users through a variety of options. In contrast, the Backup Files wizard would get minimal information from the user and then perform the task, acting more like a batch file. Like Tours, a user can access a wizard from the Help menu, the Help folder or from within a related Cairo Help topic.
- *Error Messages.* Error messages will include the error name and an understandable description of the error and/or suggestions to recover from the error. In addition, users will be able to get context-sensitive Help from within error messages, since error messages will contain a "Help" button. We will encourage authors to provide specific help content for the error messages, such as troubleshooting or diagnostic content.

Benefits

There are several benefits of this approach for user assistance.

- Users can easily get help from wherever they are or whatever they are doing. With the combination of quick context-sensitive help and more comprehensive on-line help (articles, wizards, Tours), users have a rich learning and support environment that is tailored to the users learning style.
- Users will be more likely to find the information that they need. A common complaint of users today is that they don't know what something is called and therefore have a hard time finding it with on-line help. In fact, many users resort to the Index just so they can learn the appropriate keywords. With the combination of displaying the structure of a help file for browsing purposes and content-based searching, users should be able to more easily find information that they desire. Even if the keyword is unknown to the user, their search will be over all the help contents and they can use the hierarchy as they would an index to find related articles.
- Users can find all help information provided for an area, regardless of which application supplies it, since they can search across all help files. In addition, the user can see a list of all the help files they have access to and browse the desired help file.
- MIS can customize help content as well. Annotations allow MIS to provide their users with customized versions of Help files, without altering or removing any of the content authored by ISVs. For example, MIS can customize the application's help file to include notes on company policy and resources (e.g. printers).
- The incorporation and integration of different types of help allows users to find and use the method that is most suitable for their learning needs. For example, a user can find all Tours or wizards on formatting from Cairo Help.
- Cairo Help uses and showcases the Cairo Design Environment. This presents a consistent model to the user for information browsing.

Compatibility

Downlevel help files contain the same functionality as when authored. This means that if a user prints a help article in a downlevel file, the article prints according to the WinHelp 3.x conventions (which may be different than the Cairo conventions). The only additional functionality available for downlevel files is the content indexing (which permits contents searching) and the listing of and access to help files as part of the larger list of all help files. The term "downlevel" refers to anything that ships before Cairo: Windows NT, Chicago, and Win 3.x.

Non goal

- Not a goal to provide more functionality in old help files (just existing functionality)
- It is not a goal to have this tool be the cross-platform help engine (e.g. on the Mac).

Task Assistant — Bill Mitchell

The Cairo shell provides an exceptionally powerful yet simple user interface by which system operation may be controlled manually. Cairo Task Assistant extends on this shell functionality with facilities which enable end-users, power-users, and system administrators to automate their routine tasks. Just as human office assistants handle recurrent tasks for busy business executives, so, to does Cairo Task Assistant take care of common chores for busy Cairo users.

Goals

The goal of Cairo Task Assistant is to amplify user productivity. Task Assistant is designed to empower all Cairo users with the ability to trivially automate their daily tasks, be they mail filtering, resource monitoring, file backup, or other more complex tasks.

The end-user employs Task Assistant typically by editing and reusing existing scripts from the script library shipped with Cairo. The system administrator or power user, on the other hand, typically create their own scripts from scratch. Task Assistant scripts are iconic visual programs which users create by dragging existing shell objects into Task Assistant containers. Within these special containers objects take on behaviors associated with

their various *commands* and *events* (the same commands and events that appear on objects context menus). Icons corresponding to commands on objects are graphically wired together to form intuitive dataflow diagrams.

The Task Assistant facility is comprised of a graphical editing environment and a set of Task Assistant Components which are drawn from the domain of interest. Existing shell components such as folders and files constitute legitimate Task Assistant Components, and additional components which serve no purpose except in Task Assistant Scripts are also provided. Additionally, power users and ISVs may author new Task Assistant Components in Object Basic or in lower level languages like C and C++.

In the initial release of Cairo, Task Assistant is mostly targeted towards scripting in two general problem domains: *Mail Automation* and *System Administration*. Actually, however, users can create scripts to solve problems in virtually any domain, given the appropriate domain-specific components.

Benchmark

- Apple *AppleScript*
- Metaphor *Capsules*

As Task Assistant embodies a novel object-based dataflow graph approach to scripting, it currently has no competitors. Nevertheless, a new scripting system for Apple's Macintosh system, *AppleScript* and a successful business analysis scripting system Metaphor *Capsules*, seek to accomplish many similar goals. Task Assistant compares favorably with these products in the following areas, given in order of priority:

1. Ease of script creation
2. Ease of script comprehension
3. Script language completeness and generality
4. Time and space performance

Features

- *Direct-Manipulation-Based.* Users create Task Assistant scripts using mouse drags and rubberbanding connection actions, instead of through the more traditional typing of arcane textual statements.
- *System Integration.* Existing shell level objects and their commands and events are leveraged, and anyone familiar with shell-level UI manipulations will already posses the skills necessary to manipulate scripts.
- *Controlled "Live" Environment.* Scripts may be constructed and modified while running, or execution may be disabled during script editing, allowing for a more traditional program development cycle.
- *Connect-Time Type Checking.* The script editing environment will only allow users to form connections that make sense. If the user tries to stretch a rubberband connection that, type-wise, has no valid semantics the mouse cursor is repulsed, providing immediate feedback to this effect.
- *Event Triggered and Immediate Execution.* Scripts may be stored and run later, or run immediately. They may be triggered either manually, by the user, or in response to event subsystem events.
- *Drag-Drop Activation.* Script components which accept a single input may be activated by dragging and dropping an object of the type expected by that input onto the component. This enables incremental script testing, and leverages on the users' familiarity with drag-drop operations.
- *Visual Debugging.* A debugging mode is provided which allows users to see an animation of small icons corresponding to the data elements that are being transported across connections to the various Task Assistant components.

Benefits

The Cairo graphical user interface makes file system and program management operations accessible to even the inexperienced end-user. The graphical scripting techniques embodied in Task Assistant will provide an analogous benefit in the area of task automation. System administrators, power users, and end-users alike who wish to avoid the complexities of low-level software construction, but who wish to automate some aspects of their work through

programmatic means can build Task Assistant scripts without programming. Since a single straightforward execution metaphor is employed and simple direct-manipulation visual means are used to create scripts, individual users will require little or no training to use Task Assistant. And by creating Task Assistant scripts to automate routine computer tasks, users will significantly enhance their own personal productivity. By sharing scripts (for example, workflow scripts) at an organizational level, efficiency of entire enterprises will be vastly enhanced. In addition, ISVs can easily integrate their own Task Assistant components into Cairo, just as they can add VBXs to Visual Basic. This imparts a high degree of flexibility and generality to the Task Assistant system as well.

Non goals

The Task Assistant scripting environment will not replace the need for true programming language systems like Object Basic and Visual C++. Also, it will not obviate the need for any of the Cairo programming-related infrastructures like the Component Object Runtime or the Event Subsystem. (In fact, Task Assistant leverages these system features extensively). Instead, the Task Assistant environment will serve as an ideal open framework for plugging together components using a simple, graphical means, to form useful task-automation scripts.

Component Forms — Jan Miksovsky

Goals

Component Forms provides Cairo applications with a hostable set of tools for designing and rendering custom forms. The foundation of this technology is a general-purpose form and control architecture based on CairOle interfaces. Component Forms also provides standard user interface components for form design. The first version of this technology will meet the needs of the following host applications: the InfoDoc Form Designer, the Property Sheet Designer, and the Filter Pane Designer.

Benchmark

Component Forms subsumes the performance and capacity requirements for forms in Visual Basic for Windows. A primary customer of this technology is the InfoDoc Form Designer, and Component Forms must enable it to build forms comparable to those in Lotus Notes.

Features

- *Standard design tools.* Component Forms supplies the user interface of a set of common forms design tools. These include a form design window, a Controls toolbar, and standard form and control property pages. These tools will be consistent with those in other Microsoft drawing applications (Draw, Excel, PowerPoint).
- *Extensible architecture.* A host application can extend the behavior of both the form design environment and the standard form run-time behavior. For example, an application could define new form or control properties.
- *Compound document support.* Forms can be OLE 2.0 clients as well as servers.
- *Language-independent.* A host application can bind form and control events to a programming language. The InfoDoc Form Designer, for example, allows a person to describe form behavior using Object Basic.
- *Component control architecture.* ISVs can design new custom controls for use in any Component Forms host.
- *Standard component controls.* The forms package includes a set of standard Windows controls—check boxes, option buttons, text boxes, and so on—packaged as component controls. The set also includes an addressing control and a rich text control.

Benefits

Component Forms dramatically reduces the cost of developing an application that can design and render custom forms. This lets vendors offer MIS groups and consultants better tools (such as the InfoDoc Form Designer and the

Property Sheet Designer) to build custom solutions. This in turn gives end users better tools for entering and viewing data. Because Component Forms technology sets a standard that other host applications can build upon, people who design and use forms in one application can transfer that knowledge to other applications. Also, as ISVs produce new component controls, these controls become immediately usable in all Component Forms applications

Compatibility

The properties and events on forms and controls will be consistent with those in Visual Basic.

Non-goals

Component Forms will satisfy the forms requirements of the host applications mentioned in the "Goals" section, but it is not a goal to solve additional requirements presented by other hosts. The first version of Component Forms runs only on Win32; until Cairo ships, it will not be a goal to port this technology to Win16 or the Macintosh.

Cairo Design Environment - Darren Remington

The Cairo Design Environment allows MIS-level users to turn standard Cairo folders into intelligent information sources or even custom information sharing applications. These applications focus on adding structure and organization to unstructured information. The Design Environment empowers end users with the ability to solve their own specific information sharing, information organization, and information management problems through the creation of these customized folders - called *Smart Folders*.

Goals

Enable work group scenarios. The Design Environment builds on Cairo's architecture and services to enable true distributed information sharing - including support for information gathering, organization, browsing and distribution. While the information management facilities provided by the Design Environment are optimized for groups, they are also applicable to personal information.

Empower users to solve their own business problems. The CDE empowers users to solve their own specific information problems. The Design Environment is targeted at users who are capable of writing application macros (e.g. Excel, 1-2-3) or of using any of the various flavors of Embedded Basic - VB, Word, Access, etc.

Sell Cairo The Design Environment provides unique and compelling functionality which will draw users to Cairo. Also, the CDE is the first example of a native Cairo application and showcases what applications under Cairo can do - it will set the benchmark for other Cairo applications.

Benchmark

The primary competitor for the Design Environment is Lotus Notes. It is an explicit goal to superset Notes in all the application areas it addresses.

Features

- *Completely integrated into the Cairo Shell.* Users do not have to learn new metaphors or tools. All functionality is based on extending and leveraging Cairo's objects and folders. The CDE leverages the entire set of object management facilities and the users understanding of them.
- *Custom Views.* Users can define custom views specific to their data and/or needs.
- *Explorer Customization.* Smart Folders can customize the appearance the standard Explorer menus, toolbars, filter panes, etc.

- *Folder Behavior*. Folder behavior can be customized in numerous ways including: policies (e.g. restricting the types of objects that may be stored in a folder); behaviors of objects in the folder (e.g. versioning, checkin/checkout); replication (at the folder rather than volume level); etc.
- *Forms*. Designers of smart folders can create forms for information gathering, messages, etc. Forms can also play the role of documents (called InfoDocs) in smart folders like topic and reply documents in a BBS folder, a customer report in a tracking folder or a bug report in a bug tracking folder.
- *Programmability*. Nearly all customizable aspects are also programmable using Object Basic including forms, menus, toolbars, folder policies, folder events, etc.
- *Seamless support for applications*. By supporting the standard Cairo API and services (like OLE2), applications get all the benefits of smart folders - there are no "smart folder" APIs.
- *Leverages system services*. Smart Folders leverage (seamlessly and transparently) all the standard Cairo services - replication, security, content indexing, queries, etc.
- *Help*. Provides the basis for help authoring (in conjunction with InfoDocs).
- *External Integration*. Via an ODBC provider, the CDE can access data in external databases. To the CDE, the database appears as OFS objects and properties. Conversely, ODBC databases (like MS Access) can access smart folder (OFS) data and indexes.
- *Extensible by ISVs*. Third parties can add additional views, functions and features..

Benefits

Information management issues are universal yet they defy a single pre-packaged solution. Cairo and the Design Environment provide users with the tools to solve their own information management problems according the their specific requirements.

Users are not required to work in a separate "shell" or to use services that are separate and distinct from the system services (e.g. security, directory, etc). The shell provides the additional functionality seamlessly so uses do not have to learn new tools or metaphors and, they can continue to able to use their existing applications.

To fully leverage the CDE services, application developers need only support the standard Cairo API and services (like OLE2) - special APIs, etc are not required.

Compatibility

- The use of Object Basic is consistent with other implementations throughout the company.
- The CDE uses Component Form technology.

Non Goals

1. *The Design Environment is tied to Cairo*. The possibility of runtime clients on other platforms will be considered after Cairo ships.
2. *Lotus Notes compatibility*. There are no plans for converting Notes databases or data.

Under Consideration

- Chicago may incorporate some document management functionality that we will need to be compatible with if it actually makes it in.

InfoDocs — Jan Miksovsky

Goals

InfoDocs are a general-purpose document type with which Cairo users can edit structured information through a visual form. Examples of InfoDocs are mail messages, help topics, and bulletin board articles. Although they can exist anywhere, users most often store them in folders that can exploit structural knowledge of the InfoDocs they contain to facilitate browsing, querying, and analysis. For example, if a folder contains mail messages, then someone can construct a view for that folder that shows properties unique to mail messages, thereby making it easier for other users to find mail messages with the Explorer.

Cairo customers can create new InfoDoc Forms using the InfoDoc Form Designer. This extends the Component Forms design tools by providing: a model for how form definitions are stored, the ability to program form behavior using Visual Basic for Applications (the technology known internally as Object Basic), and a system that automatically loads and saves data between corresponding controls on a form and properties in an InfoDoc. The InfoDoc Form Designer will be able to build the forms in the sample Cairo applications such as Mail and Help.

Benchmark

InfoDoc Forms must be equivalent to, or more powerful than, Lotus Notes 3.0 forms in visual and behavior expressiveness. InfoDoc Forms must also be at least as functional as the Note form in Microsoft Mail 3.0.

Features

- *Custom form design.* The InfoDoc Form Designer is a host application for the full-featured Component Forms design tools, which provides: a standard drawing model, a Controls toolbar, and editing of properties on forms and controls (including QuickInfo help text). Designers can create simple forms without having to write code.
- *Automatic control-to-property binding.* When a user opens a document, the form controls are automatically filled with the values of the corresponding document properties. Likewise, control contents are saved in document properties when saving. These document properties are visible in places such as folder views.
- *Menu bar and toolbar designer.* The form designer can create a customized menu bar and toolbar.
- *Basic programmability.* The form designer can use Visual Basic for Applications to define the behavior of forms, controls, menus, and toolbar buttons. Designers can write code for tasks such as calculating field values, enabling/disabling controls when appropriate, and verifying that the user has entered valid field values. Through Basic, the form designer can take advantage of OLE Automation to programmatically access and manipulate objects from other products (e.g., Excel spreadsheets).
- *Standard editing features.* Including Find and Replace and spell-checking.
- *Rich text.* The rich text control (from the standard Component Forms control set) supplies: typeface support, paragraph formatting, bullets, tables, OLE 2.0 embedding, and hyperlinks (à la Windows Help). Users can import rich text from RTF, Microsoft Write (.WRI), and Microsoft Word (T3) files.

Benefits

The presence of a complete form design environment encourages Cairo designers (ISVs, consultants, and in-house development teams) to produce forms for Cairo applications. By visually organizing document contents on a form, a form designer simplifies data entry and retrieval; users know what information they need to enter, and the form can assist them by calculating and verifying field values. By structuring the document contents as file properties, the form designer facilitates the browsing, querying, and analyzing of entire sets of documents in folders. For example, users can see InfoDoc properties as columns in views.

Non-goals

The InfoDoc Form Designer specifically creates document editing applications; it is not a goal for it to build the range of applications possible with the Visual Basic development environment (games, database front-ends, etc.).

Basic and OLE Automation — Jan Miksovsky

Goals

Cairo will promote Visual Basic for Applications (previously known as Object Basic) and OLE Automation (the public term for IDispatch) as key elements in making the system programmable. Visual Basic for Applications is the next evolution of the Basic language being developed by APPU; an application can host this language to provide Basic programmability as a feature. OLE Automation is an OLE 2.0 technology with which one application can directly access and manipulate the objects of another application. For example, the Basic code for an InfoDoc Form can programmatically change the contents of an embedded Excel spreadsheet.

Benchmark

AppleScript and AppleEvents from Apple supply a somewhat comparable cross-application scripting technology, but there are no specific performance or capability requirements vis-a-vis these products.

Features

- *Basic run-time*. The Cairo desktop and server packages include the files required by Basic host applications.
- *Basic host applications*. The InfoDoc Form Designer and the Type Designer that ship with Cairo are both programmable with Visual Basic for Applications.
- *Programmable system objects*. The Cairo shell exposes many of its own objects through Basic type libraries or OLE Automation. Basic programmers will be able to manipulate files, folders, the desktop, the tray, etc.

Benefits

Supporting Visual Basic for Applications in Cairo promotes it as a language standard. The VBA language is extremely powerful, and enables the construction of complex applications that solve real customer needs. Moreover, a Cairo designer (e.g., an in-house developer) who has used VBA in an application such as Excel can leverage that knowledge when using other host applications such as the InfoDoc Form Designer. Lastly, through Basic type libraries and OLE Automation, designers will have easy access to a full range of Cairo system objects.

Under consideration

We are considering building a general-purpose host application with which users could write cross-application macros in Basic. (This would be similar to the VBasic application planned for Chicago.)

Cairo Email - John Tippett

Cairo is the first version of Windows to have messaging designed in as an integral part of the operating system interface and architecture. This section describes the mail client for Cairo.

Goals

Cairo Mail integrates powerful end-user email features with the Cairo user interface and system components. The result is a system which is easy-to-use and allows users to more effectively integrate messaging objects with other documents on their system.

Benchmark

Apple's OCE and AppleMail are the direct competitors of Cairo's messaging system. Cairo messaging includes all the functionality of current Microsoft Mail products.

Features

- *Shell Integration.* Cairo messaging is completely integrated with the Cairo shell, eliminating the need for a special purpose electronic mail client and exploiting native Cairo user interface components, including templates, the Explorer Scope and Filter panes, and links.
- *System Integration.* Messaging is integrated with the Cairo system services, so users and administrators do not have to have separate directory services, security schemes, and directory structures for mail and the resst of their system.
- *Forms.* Cairo users can send different forms as email messages, and can create new message forms.
- *Object Transfer.* In Cairo, users can send any file system object to another user without having to bundle it up within a mail message. This allows them to use the messaging system to perform a "disconnected copy" operation and lets them send objects directly without adding them as attachments to some other mail document. Applications can also use this ability to provide services which require application-to-application store-and-forward capabilities..
- *Filtering.* Users will be able to automatically organize incoming email based on customizable filters.
- *Remote Use.* The mail client can be used while not connected to the network and users can access their mail by dialing-up to the network with a modem.
- *Viewers.* Cairo Email will provide the users with several useful email-specific views, including a threaded conversation view
- *Fax support.* Users will be able to easily send faxes with the Cairo Email client.

Benefits

Cairo makes messaging simple for the user by integrating it completely with the operating system. In Cairo, there is no need for a specialized electronic mail client; mail can be viewed directly from the ***Explorer***. Similarly, because messaging objects are truly Cairo system objects, users can combine messaging features and email messages with their other tasks in ways which make them more productive.

Compatibility

The Cairo Email client is fully MAPI compliant; it will work with all MAPI service providers and directory services.

Cairo users can interoperate with non-Microsoft electronic mail systems via gateways.

Users who move to Cairo from other leading electronic mail systems can import their existing mail files into Cairo.

Non goals

Under consideration

Cairo Email may provide a serial routing form.

Improved system setup - Bob Muglia

Cairo will provide an integrated setup program which makes it extremely easy for a first-time Windows user to install the system while at the same time providing the flexibility to allow experienced users to customize their system.

Goals

Setup is the first experience a user has with Cairo. The primary goal of setup is to correctly install the system while keeping user interaction to an absolute minimum. A secondary goal is to provide the flexibility a sophisticated user demands.

Benchmark

Cairo should be as easy to setup as the Chicago release of MS-DOS based Windows.

Features

- *Complete operating system installation.* Cairo setup will prepare the computer for operating system installation, including hard disk setup and network installation. In order to provide complete installation, Cairo setup will be broken into two parts: textual and graphical.
- *Plug and Play.* Cairo setup will utilize the Plug and Play technology originally developed for Chicago to simplify installation. Computers and peripherals which correspond to the Plug and Play spec. will be setup with no user intervention.
- *Express and Custom* In the tradition of Windows setup programs, both express and custom options will be offered.
- *Push install* Setup will utilize the features of Windows NT 3.1 to support upgrades from Windows NT to Cairo without touching the workstation.
- *Machine-profile setup* Similar to Windows NT, machine-profile setup will allow an administrator to "snapshot" an installation, including hardware and application setup and then clone that installation on other computers.
- *Windows 3.x upgrade.* Cairo setup will upgrade a Windows 3.1, WFW, or Windows NT workstation and will convert the users program groups, .ini files, and registry to Cairo formats.
- *Chicago upgrade.* Cairo setup will upgrade a Chicago system, retaining all shell, system, and application settings.
- *FAT to OFS conversion.* Setup will support conversion of FAT partitions to OFS. Conversion of DOS 6 compressed partitions to OFS will also be supported.
- *NTFS to OFS.* Setup will support conversion of NTFS partitions to OFS.
- *Non-destructive re-partitioning.* Setup will allow the user to re-partition their disk without destroying data, thus freeing-up space for an OFS partition while keeping some files on FAT for multi-booting. It will also be possible to use this re-partitioning utility to remove OFS from their machine, thus allowing the user to go back to MS-DOS Windows and FAT.
- *Cairo desktop to server upgrade.* The Cairo server box will include a setup option which supports an upgrade from the Cairo desktop to the Cairo server, preserving all configuration information.
- *Elimination of Maintenance-mode setup.* Cairo no longer supports maintenance-mode setup. Once installed, all Cairo configuration including hardware installation is performed in the control panel.
- *Removal of previous versions of MS-DOS and Windows.* Cairo will provide a utility to remove previous versions of Windows, Windows NT, and MS-DOS. This allows a user who has chosen to dual-boot between

MS-DOS and Cairo to easily remove MS-DOS from their system once they are comfortable that they no longer need this environment.

Benefits

Cairo setup provides for very simple installation while retaining the flexibility of customization. The upgrade options allow the user to retain previous Windows configurations in their new Cairo environment, thus making the transition to Cairo easier. Upgrade from a previous version of Windows NT can be performed without touching the workstation, dramatically reducing the upgrade cost. Machine-profile setup provides a straight-forward way to clone a computer configuration with an almost questionless install. The OFS conversion utilities and support for re-partitioning minimize the cost associated with moving to a new on-disk format.

Compatibility

The Cairo setup program will be modeled after Chicago, thus profiting from the design work and user familiarity of the MS-DOS based Windows platform.

Non goal

HPFS to OFS conversion is not supported. It will, however be possible to upgrade an HPFS disk to OFS by first upgrading it to NTFS and then performing an NTFS to OFS upgrade.

We will not support OFS to FAT conversion. If the user wishes to remove Cairo from their system, they can copy relevant data stored on OFS back to FAT (with corresponding loss of OFS unique storage such as properties) and use the re-partition utility to remove the OFS partition.

IV. Requirements: Object Infrastructure

Object Programming Model - Edward Jung

Goals

This feature provides low-level ISVs with guidelines of how to structure their applications to provide and use Cairo object services. It's goals are to provide a structure to system programming, to extend and leverage OLE 2, to enable programming using ordinary languages like C and Cobol as well as object-based languages such as C++, to assist in debugging component applications, and to assist class and code library vendors in structuring their libraries according to a well-defined set of system services. It is also a principal goal to deploy the same object model on all Microsoft systems platforms, from desktop systems to television-based computers, from personal digital assistants to telephones.

Benchmark

Main competitors include:

- NeXT Computer's NeXTstep
- Apple Computer's Bedrock and Exemplar
- Taligent's Object Framework
- Object Management Group's Object Management Architecture

Benchmark areas:

- Migration path from existing 32-bit OLE 2 applications

- Enables high-performance component-based software
- Ease of understanding by programmers
- Amenable to implementation by any language vendor

It is a goal to engage in open process and usability feedback from early ISVs in the programming model to improve our performance in the benchmark areas.

Features

Object model features

- *Support for extensible naming.*
- *Support for extensible object types and polymorphism.*
- *Type-safe run-time binding.*
- *Distributed activation and binding.*
- *Programmer support features.*
- *Run-time type information.*
- *Language independence.* (small and large objects)
- *System classes.*
- *Error management.*
- *Object definition language*

Run-time environment features

- *OLE 2 compatibility*
- *Rich set of predefined interfaces for interoperability.*
- *Support for component programming.*
- *Interoperability with rest of system.* (also error management)

Benefits

Many of these benefits are for the programmer rather than the end user, so the end user is only impacted indirectly, i.e. with better software:

Leverage of services -- By having a single object-based programming model in the system, Microsoft systems platforms can achieve a high-degree of interoperability between all objects, whether they are documents, multimedia clips, programs, or services. This kind of interoperability is a key facilitator for the widespread, distributed, interconnected, ubiquitous information-based vision known as Information at Your Fingertips™. This will give ISVs a single conceptual programming model for the huge variety of Microsoft platforms, and give their software an unprecedented level of interoperability; for example, accessing a server-based information service from a personal digital assistant or television.

Service polymorphism -- The architecture enables open competition between providers of services. This is fundamental to the Windows Open Services Architecture and stimulating software providers while maximizing the choices of software users.

Extensible and modular system -- The programming model allows the system to be modular. This enables more configuration flexibility to meet customer needs. It also means less product needs to be shipped to revise the system in the future, much like the way Apple's System 7 is upgraded through modules rather than through monolithic releases.

Interface-based interoperability -- Basing the system on interfaces helps define standards for interoperability. This eventually results in more software that works together, rather than having software defining islands of operation and forcing unnecessary purchasing restrictions, e.g. everything must be compatible with Lotus 1-2-3 or Microsoft Excel.

Enables components and containers -- Components and containers (described elsewhere) are useful structuring mechanisms used to make the system easier to use, to encourage interoperable software, and to simplify the user interface. The object system is what allows these things to be built.

Easier to understand system (system has structure) -- The use of a class hierarchy and other object oriented organization principles offer guidelines to programmers and users to how the software needs to be structured to "play well" with the system. This makes programming easier because the responsibilities of software are more strictly defined. This also makes life for the user better because software behaves in expected ways.

Offers a basis for IS-level methodologies -- The object system allows traditional 3rd generation systems to participate in the system through objects. This allows a graceful migration of Information Systems (IS) methodologies to software development. The Cairo object system will be used by our enterprise strategy for this purpose.

Compatibility

The model is compatible with OLE 2 component object model and the component architecture for controls. The component architecture will be used by visual basic and object basic, making the model compatible with those products as well in the future.

The model is somewhat compatible with C++. The C++ language has a number of features that conflict with the object model, prohibiting complete compatibility.

The model is compatible with existing C++ language tools.

The model is different from the Win32 API, although they are not incompatible. Wrapper objects can have extractable handles as needed (examples are OFS and controls).

Non goal

The deliverables of the programming model do not have to explicitly support tiny application-internal objects such as spreadsheet cells, although the programming model makes any such supporting code, i.e. installed by an application or code library, transparent to the system.

The programming model does not have to provide a class or code library, only enable them to be written.

The programming model does not have to support C++ at any "fancy" level.

Under consideration

Lower level support for bi-directional connections for the component model. More explicit support for named interfaces

---

Component Object Runtime (COR) - David Stutz

In order to be generically useful, pluggable software components must be able to describe both the services which they are designed to provide and the services that they require from other components. Cairo will express this combination of services and requirements in a generic way by representing them as *software connectors*.

Objects are the primary organizing feature of Cairo, and building new objects by composing existing objects together is an activity that *everyone* who uses Cairo will participate in. Elements that can be composed together using connectors will be called *components*. These are areas within Cairo that will exploit COR:

- Forms and controls
- Explorer views and filter panes
- Shell property pages, commands, and events

Goals

The component object runtime (COR) is a lightweight extension to the OLE component object model. It will provide a standard mechanism that objects can use to describe their structure, their binding characteristics, and persistent binding information.

Developing components for Cairo should be as easy as possible. To this end, COR will provide a framework for managing the routine bookkeeping that is currently done manually by all OLE programmers when binding to object interfaces. It will also handle storage of property values, type information, interface IDs and monikers.

Often, object compositions built by joining software connectors together need to support both persistence and self-description. Persistence will be supported by representing a component's persistent state as properties; a system implemented property stream mechanism can then be used to save and reload this state. Any component will be able to save its connectors as part of this property state – as a result, COR will re-establish the bindings represented by these connectors whenever a component is loaded into memory. For self-description, COR will provide an easy-to-use standard implementation of the existing OLE type library interfaces.

Benchmark

NeXT Interface Builder

Features

- *Lightweight component runtime* - system-provided interface implementations for components
- *User interface components* - Windows, property sheets, Cairo-specific view panes, and controls
- *Forms editor* - Environment for composing user interface components
- *Generic container editor* - Environment for composing components that have no visual representation
- *Type information* - Tool support for creating object descriptions at either runtime or compile time
- *Build/project management* - Tailored makefiles for producing Cairo klass DLLs and containers

Benefits

- *Ease of use* - Programmers and end-users who do not wish to understand system-level Cairo programming can access, manipulate, and store components from domain-specific environments. No understanding of the mechanics of using interfaces or the system binder need be required.
- *Enhanced productivity* - COR can be used to package code for re-use. This kind of packaging enables programmers to concentrate their efforts on producing code that solves new problems, rather than on reinventing the wheel
- *Metaphor for extensibility* - Components are easily understood as "parts" of the system that can be replaced or added. This metaphor leverages the capacity for polymorphism already present in the object model.
- *Conformity* - Pre-fabricated components reinforce "correct" Microsoft usage. They augment system documentation as interactive and accessible examples.
- *Ease of customization* - Service providers can use connectors to provide "hardened" component interfaces for their program elements. Application builders can provide customized look-and-feel for those services, without disturbing core technologies, by selectively replacing program elements.
- *Language neutrality* - Components do not assume a language environment. They can be used from and by any language that supports binding to Cairo interfaces. They can peacefully coexist with application frameworks.
- *Enabling technology* - Tools vendors can rely on the services of COR to provide the information needed to produce very generic development components and environments.
- *Interoperability* - Connectors are a natural way to both define and enforce the correct usage of interoperability standards such as ODBC or MAPI.

- *Streamlined deployment* - Component-based software development can be used to produce effective prototypes rapidly. Prototypes can be smoothly converted into production code by incrementally optimizing, replacing, or bullet-proofing the components used initially.

Non goals

COR does not intend to describe a new universe; it merely describes a way of organizing and automating existing OLE infrastructure. It is a thin veneer on top of the component object model, and can be ignored by those who choose not to use its facilities.

Although Cairo may provide visual programming facilities for its users, the features described here have no support for control flow or sequencing, and should not be considered to be visual programming.

Under consideration

- *WOSA components* - ODBC and MAPI objects sets
- *ADT components* - list, hashtable, collection, trees, etc.
- *Instrumentation components* - dials, gauges, and meters

File Systems - Nikhil Joshi

The Object File System (OFS) for Cairo provides advanced persistent storage for objects. It is a robust, recoverable file system that provides direct support for object properties, in addition to content indexing, content and property based queries, link tracking, and the basis for replication and the distributed file system (DFS).

Cairo can also use other file systems such as FAT, NTFS and HPFS. However, the user and administrator should understand that these other file systems are not optimized for storing Cairo objects, and much Cairo functionality will be lost if OFS is not used.

Multiple file systems can be used with Cairo. For example, a particular Cairo installation may have two file systems, one FAT and the other OFS. They are plugged into the system using the IFS (installable file system) mechanism in NT. An OSH (object store handler) written for OFS and non-OFS file systems is responsible for interfacing the file system to Cairo.

Users can also convert their existing file systems. Cairo will provide FAT to OFS and NTFS to OFS conversion. HPFS users can convert to OFS, by first converting the HPFS volume to NTFS using NT's conversion utility, and then converting the NTFS volume to OFS.

The remainder of this section discusses the Object File System, the recommended persistent store for Cairo.

Goals

OFS provides high performance persistent storage of objects and native file system support for the OLE2 storage interfaces. OFS is the file system of choice for Cairo.

Benchmark

OFS will provide similar if not better performance than FAT.

Features

- *Administration.* OFS will notify the administrator of interesting file system events, such as quota violation, and read/write errors. OFS will support monitoring using the meters, counters and other measurement tools provided by System Management.
- *Capacity.* OFS can address volumes as large as 2 terabytes. OFS will use the same compression algorithm as DOS 6.0.

- *Content Index Notification*. OFS notifies the Content Index when objects have been modified so that it can update itself.
- *Dynamic Repartitioning*. An existing FAT partition can be "shrunk" without loss of user data in order to create a new partition for an OFS volume.
- *Efficiency*. OFS uses an advanced allocation algorithm to provide efficient storage for small objects with as little as a few hundred bytes of overhead per object. OFS provides copy-on-write support. Copies of objects share parts of streams that are identical between them, minimizing storage requirements.
- *Filesystem Conversion*. OFS provides a utility to convert FAT and NTFS volumes to OFS with no loss of user data even in the event of a power failure.
- *Junction Points*. OFS catalogs can be joined together through the use of junction points. OFS volumes can serve as non-leaf nodes in the distributed filesystem whereas non-OFS volumes can only be leaf nodes.
- *Link Tracking*. OFS provides support for link tracking across volumes.
- *Long Filenames*. OFS allows long, Unicode filenames, and mapping between 8.3 naming and long filenames identical to NTFS.
- *Multi-spindle Volumes*. OFS volumes can be composed of multiple physical disk drives. Extra drives can be easily added to increase disk space as needed. Total volume size, regardless of number and sizes of individual spindles cannot exceed two terabytes.
- *OLE2 Support*. OFS implements the OLE2 IStorage and IStream interfaces directly, eliminating the need for Docfiles, and thus significantly improving performance and efficiency. Multistream OLE2 objects are really composed of separate multiple streams under OFS.
- *Property Support* OFS implements the property interface for efficient access to properties. Speed of property based queries is enhanced by using a property value index for frequently accessed properties.
- *Query Support*. OFS resolves both content based queries, and queries on object properties. Queries can span multiple OFS volumes.
- *Quotas*. OFS allows the administrator to assign disk space quotas for users.
- *Recoverability*. OFS maintains a log of all directory/allocation changes made to the disk to permit fast recovery in the event of a system failure. RAID, disk mirroring and striping are supported by layering device drivers provided by NT for these functions.
- *Remote Access* OFS will extend the Server Message Block (SMB) operations to allow use of new OFS features across the network.
- *Replication*. OFS supports object replication by maintaining deletion logs and other data structures used by the DFS and DS.
- *Security*. OFS provides ACL based security for every object.
- *Summary Catalogs*. OFS provides efficient storage of summary catalogs and uses its content and property value indexing technology to enhance queries directed against summary catalogs.

Benefits

OFS provides a high performance, robust file system customized to support the advanced features of Windows Cairo. It uses a new method of allocating disk space that can store many small files (which will be common under Cairo) without wasting disk space even on huge volumes. If the user is running out of disk space, he can simply add a new disk to the system and grow the amount of available disk space that OFS manages without backing up or moving any existing files. An OFS volume can be grown like this up to two terabytes. OFS groups small properties together so that a minimum number of disk accesses are required to fetch the values - often as few as a single disk hit.

OFS maintains a log of all its pending disk modifications so that it knows what actions have to be redone, and what have to be undone to recover in the unlikely event of a crash. More advanced fault tolerance features such as RAID, disk mirroring and disk striping can be used by adding the appropriate device driver to NT. OFS can work transparently with these features.

OFS implements the query interface on top of the file system so that users can ask for information about the objects that OFS maintains on its volumes. To enhance this functionality, OFS maintains a content and property index. Users won't have to periodically execute an indexing program and tie up their system while it runs. The indexing engine runs in the background, allowing the user to work on their document in the foreground, confident that its content and properties will be indexed and accessible via queries.

OFS provides the basic filesystem services used by the Directory Service and Distributed File System to create a single global namespace for the user. The user no longer needs to keep track of which objects are located on which machine - OFS and the DFS keep track of this information.

All of this is provided, without sacrificing compatibility with FAT. Existing DOS and Windows programs will be able to access all existing documents. OFS provides easy migration from FAT and NTFS using the OFS conversion utility. This utility allows the user to convert their existing filesystem to OFS in place without requiring any backup or re-arrangement of files.

Compatibility

OFS will support all DOS and Win32 file APIs, both locally and remote. OFS data structures are not binary compatible with FAT so utilities that expect FAT data structures on disk will not work (e.g. Norton Utilities for DOS). OFS will convert between docfiles and the more efficient multi-stream format used by OFS as needed by downlevel and Cairo aware applications. OFS will provide a utility to convert FAT and NTFS volumes to OFS with no loss of user data.

Non Goals

- OFS will not be available on DOS or Chicago.
- OFS will not convert Netware or HPFS volumes.
- Conversion of OFS volumes back to FAT or NTFS will not be supported.
- There is no robustness support for multi-spindle volumes aside from standard logging.
- OFS does not support POSIX, and thus, will not support traversal permission, case sensitive names or equal instance hard links.
- OFS will not provide support for sparse stream storage.
- OFS will not provide EA support.

Open Issues

- What are Delphi's indexing requirements? Will they affect summary catalogs?
- How will OFS work with floppy drives? The user should not have to worry about file format changes (docfile, MSF, etc.) when copying objects to floppies.
- Do we need ACLs on embeddings or just property sets? What are Delphi's requirements?

Content based document queries and indexing - Daniel Conde

Windows Cairo includes the ability to index and search documents based on their contents.

Goals

To create the underpinnings to allow document queries. The system will quickly and unobtrusively index documents, and quickly locate documents.

Benchmark

Various commercial products provide similar features on PCs. These include Folio's Views, Larson-Davis Information Systems' InfoQue, Odyssey Development's Isys, Virginia Systems' Sonar, and Zylab's Zylndex, Saros Mezzanine and Lotus Development's Magellan. On larger systems, we have Verity's Topic, Fulcrum, BRS/Search, CCA TextView, CPI Status, and Excalibur's PixTex/EFS. Few commercial PC based systems have system level (not user application) support for this type of feature. The industry observers call this area by many names, including Electronic Document Management Systems (EDMS), Full-Text Retrieval or FTR (IDC Reports). IDC publishes a report on this.

Features

- *Fast queries* The system will index document contents and properties, rather than search through them on demand. The query runs in the kernel thus minimizing data transfer and improving performance.
- *Content & Property queries.* The system indexes the content of the documents as well as properties of objects, such as author, create-date, etc. Embedded objects can be indexed provided content filters are available for their classes. The admin may exclude/include document types to index.
- *Index Compression.* The system compresses the index.
- *Summary Catalog Support.* The system can place document indices into a summary catalog.
- *Automatic indexing.* The system indexes the documents as a background process. This eliminates the need for users to manually re-index the directories.
- *Integrated query.* Users invoke the Explorer to query for documents. This eliminates the need for a separate document retrieval program.
- *Boolean Queries* Users may search for documents using that satisfies queries based on logical queries and content queries. For example, you can ask for documents that contain the words "United States" AND "Japan."
- *Proximity searching* Users may search for documents that contain different words that are not located next to each other. This allows users to search for the word "little" near the word "book" and find a document that contains the phrase "little red book." There is no threshold for the proximity so all instances of words near each other will be returned and ranked by "closeness".
- *"Fuzzy" matching* The system will allow the user to specify how carefully it should try to match words specified in the query. The user could request only exact matches should be returned, or use wildcards like in DOS to find a range of words. The system currently supports full regular expression parsing. We will also include linguistic engines so the system can find different forms of words using inflection, so when the user asks for the word "find", we can also return documents that contain the word "found". In addtion, the system will be able to look for any synonyms for the words specified in the query. We will use the same linguistic software to optionally spell-check the user's query requests. This implies that a spelling verifier and thesaurus will be bundled into the system software.
- *Match highlighting.* Applications can cooperate with the query system to highlight query matches within their documents. The system basically executes the query with full fuzzy matching capabilities and asks the applications to highlight the words it finds inside the document.
- *Ranking.* The system can rank query hits and uses a variety of different ranking metrics.
- *Document format support.* Users can install different filters that understand a document's format in order to extract the words and follow embedded objects to index words and their locations within the document. Microsoft will ship Cairo with bundled filters for the most popular applications. We will encourage ISV's to supply format filters with their Cairo applications.

Benefits

The end users need not rely on a file name to locate documents. End users do not need to learn a separate document retrieval program since this system is integrated into the file system. Since Summary Catalogs can

contain content indexes, users can post content-based queries against remote documents without excessive network use.

Compatibility

The content index system will use filters to read the most common data formats.

Non goal

We do not index documents by parsing the content's sentences, and storing their semantic information in the index. We will not distinguish between the use of the word "crane" as a bird or as construction equipment.

Under consideration

We are investigating how sophisticated the linguistic analysis and thesaurus software needs to be.

We are investigating how to deal with documents written in multiple languages.

We are considering how future systems can index morphological lemmas or baseforms in addition to, or instead of the literal strings. This will reduce the need for morphological generation during queries, but will require more processing during index time. Such a scheme may set a better architecture for more sophisticated based queries, as we incorporate more NLP functionality.

V. Requirements: Distributed Infrastructure

Distributed Filesystem (DFS) - Aaron Contorer

This feature allows a Cairo machine to see all of the file storage volumes on the network as a single large volume, with one tree-structured namespace under a single drive letter. Another name for this feature is DOFS (distributed object filesystem).

Goals

- Make accessing a network of heterogeneous storage as easy as accessing a single local disk drive. Impose a meaningful organization over a possibly complex set of remote storage devices. Transparently connect to network drives, including the transparent use of multiple redirectors.
- Allow administrators to configure and reconfigure distributed network storage, transparent to end users of that storage.
- Transparently select, connect to, and use replicated OFS volumes without requiring end users to be aware of replication.
- Compatibly provide all this functionality to unmodified pre-Cairo applications.

Benchmark

Sun's Network File System (NFS) is a popular basic DFS that has been shipping for years.

Carnegie-Mellon University's Andrew Filesystem and Coda Filesystem, and the OSF DFS based on Andrew, are DFS's with very clever cacheing facilities to reduce network traffic.

The Deceit Filesystem (a superset of NFS) is a DFS produced in an academic environment (not commercial quality) that provides fairly sophisticated replication features.

Microsoft LAN Manager, WfW, and Windows NT 3.1 all provide access to a big set of remote disks via UNC names (e.g., "\\server\share"). However, UNC naming has never been integrated into the UI nor API of Windows

enough to appear as a DFS, and it is not as flexible as a DFS is normally expected to be (e.g. UNC name of a file must include NetBIOS name of the server that holds it).

Features

- *Single namespace.* The DFS incorporates any number of file volumes from any number of servers into a single, tree-structured namespace.
- *Installable filesystem.* Part of the DFS is implemented as an installable filesystem driver, so it is accessible from any level of the system: NT API, Win32 API, Cairo interfaces, DOS API, or other NT subsystems.
- *Heterogeneous volumes support.* Although the DFS may rely on features provided by OFS to manage the upper portions of the namespace, non-OFS volumes can be included in the DFS namespace as well. Any type of storage which is accessible as an installable filesystem can be included. At a minimum we will support all file volumes accessible from an SMB redirector and a Novell redirector.
- *Support for replicated volumes.* Any volume in the DFS may be a replicated OFS volume. The DFS automatically selects and connects to to an appropriate replica of any replicated volume.
- *Path caching.* When resolving pathnames, the DFS uses a local cache of paths and network addresses to reduce network traffic.
- *Backup support.* The DFS can be backed up (e.g. to tape) and restored, completely or partially, with files automatically being restored to the volume from which they came.
- *Storage reconfigurability.* With some effort by the administrator, the DFS allows splitting a set of files from one (presumably overcrowded) volume onto two volumes, or the joining of a parent and child volume into a single volume, or the moving of a volume from one machine to another, all transparent to the end users of these files.
- *Distributed queries.* Given permission, a user can make a query which spans multiple volumes of the DFS; the fact that multiple volumes were involved in answering the query is transparent to the user.
- *Catalog support.* Cairo creates and maintains configurable *catalogs:* centralized lists of files, automatically updated in the background. These can be used to answer queries quickly, without visiting all the different machines that hold the files.
- *Security* Volumes which provide for security (e.g. OFS volumes and NTFS volumes) still provide the same levels of security when in the DFS. Therefore the DFS supports NT ACLs and privileges such as Backup Operator. The DFS supports Kerberos authentication (see "Security" section of this document for details on Kerberos in Cairo).

Benefits

Users can access files without caring about server names, share names, or various network operating systems. All these details are hidden from the user, who sees only the namespace. Directories in the namespace can have sensible, appropriate names independent of server name, share name, or network operating system. Catalogs let users make queries that span a large number of volumes, yet still provide answers with high performance and low network traffic.

Users do not have to know about the concept of "attaching" network drives to a local drive letter. All network drives are accessible in the DFS namespace under a single drive letter, with no additional work by the user. Browsing and query thus become much simpler.

Administrators can locate files on whatever servers have space available, and if storage needs change, administrators can relocate these files without affecting users.

Replica selection provides higher performance by distributing multiple clients across multiple copies of a volume, and it provides higher reliability by connecting clients to an available copy even if some of the other copies are offline.

Compatibility

Cairo DFS uses an enhanced version of SMB (the Server Message Block protocol), so it cannot be accessed by any pre-Cairo clients without a new redirector. Chicago product 1.1 will include such a redirector as a standard feature, so Chicago 1.1 clients will be able to use the Cairo DFS. A DFS-capable redirector will not be provided for any other pre-Cairo platforms. However, the individual disk volumes which make up the DFS can still be accessed by downlevel clients as individual sharepoints (assuming the server's owner creates said sharepoints).

Cairo DFS exports the same API that OFS exports. (In turn, that is roughly a superset of the API that NTFS exports.) Cairo DFS also exports some management and internal-use APIs which are unique. Cairo DFS does not use the same wire protocols as any other DFS.

As mentioned above, Cairo DFS may incorporate existing non-Cairo shared disk volumes into the DFS namespace. Cairo clients access these non-Cairo volumes using the volumes' native APIs and transports. This works even if the foreign volume is really a different filesystem (e.g. Novell) or another manufacturer's distributed filesystem

Non goal

The DFS does not need to support a configuration where there are no OFS volumes; the root volume may be required to be OFS.

Storage reconfigurability (discussed above) is not automated. It requires some work by the administrator, and if errors occur during reconfiguration the administrator may need to take some manual steps to assist in cleanup.

DFS does not support inherited ACLs across volumes in the DFS. Each volume's ACLs are handled independently just as they are today

---

Directory Service (DS) - Aaron Contorer

This is the distributed database used by Cairo to publish, discover, select, and locate network resources.

Goals

The DS provides a means for resources of all types (e.g. users, printers, disk volumes, groups, domains) to be created and published so that all machines in the Cairo distributed system can discover and use them. It provides for interoperability with heterogeneous other directory services, so that Cairo machines can discover and ask to use arbitrary foreign resources as easily as Cairo-provided resources.

Benchmark

CCITT X.500 Directory Services spec (1988 and 1992 editions) is the standard which most DS manufacturers wish to approximate, though virtually nobody wants to follow X.500 exactly due to vague definitions, inefficient protocols, and hard-to-use APIs.

Novell NetWare 4.0 Directory Service is likely to be a *de facto* standard by the time Cairo ships, and has a large subset of the features that Cairo DS has. It is likely to be the main competition. It is scheduled to ship around March 1993.

Banyan StreetTalk and StreetTalk Directory Assistance were the most widely respected and most functional directory service for PC networks as of 1992. Their very simple design has not prevented them from being very useful.

Unix White Pages and Yellow Pages (aka Network Naming Service) offer a small subset of what Cairo DS needs to offer, and are gradually becoming outdated.

OSF DCE Directory Service is a two-level thing: the upper level is X.500 (see above); the lower part is the Cell Directory Service (CDS), which is shipping today and makes a good RPC name service. (One version of CDS is also known as DEC DNS.)

Features

- *Integration with Cairo.* The Cairo DS imposes minimal overhead on the system, the programmer, the administrator, and the user, because the DS is simply a use of Cairo itself, not a separate service. DS objects are OFS files and can be manipulated with ordinary file APIs and Cairo storage interfaces. DS query is OFS query. DS wire protocol is the OFS version of Server Message Block (SMB) protocol.
- *MAPI address book.* Cairo DS also exposes full MAPI 1.0 address book functionality, for compatibility with mail client applications. It is also possible to build MAPI address book providers on downlevel client systems that talk to the Cairo DS, although the Cairo group is not building these.
- *DFS features.* Cairo DS uses the Cairo Distributed Filesystem to store its data. Therefore the features provided by DFS can be used not only on users' files but also on DS objects (e.g. users, printers, etc.). A few of the key features are: Hierarchical namespace. Each object in the namespace can have many properties (thanks to OFS). Queries are supported.
- *Same namespace for files and services.* Because DS objects live in the DFS, the user can browse a single namespace to find both files and other types of resources.
- *DS objects cached on workstations.* Objects representing services appear not only on the domain controller, but also on the machine that provides the service. This allows machines to access DS objects describing their own local services even when they are off the network.
- *Support for email.* Cairo DS provides a superset of the major features of the Touchdown directory service being worked on by Microsoft Workgroup Applications. This allows the Touchdown message transfer agent (mail server) to be ported easily to Cairo.
- *Global query support* Cairo DS maintains a catalog called the *global catalog*, which describes most or all of the DS objects from the entire Cairo distributed system in a single list. Thus queries such as "show me all the printers" or "show me all the users named Bill" can be answered quickly and efficiently. The global catalog can be replicated to ensure its availability at all sites on the network.
- *Locally stored address books.* Workstations can hold a local copy of a subset of the global catalog, e.g. a list of all users and their email addresses.
- *Publication of Cairo resources.* Cairo DS provides standard and simple means for any Cairo service in the network to give itself a name and describe itself. Once in the DS, the resource can be accessed by name or queried for.
- *Publication of foreign resources.* With assistance from other tools or administrators, Cairo DS can create listings for resources published by non-Cairo machines. Once listed, these resources can be found and used just like Cairo resources. The global catalog can also be configured to include objects which are manually created by the administrator -- e.g., entries representing people who do not have computer accounts but whose names and phone numbers are of interest.
- *Junction with other directory services.* Cairo DS allows the creation of *junction points* to incorporate other manufacturers' entire directory services as part of the DS namespace -- in effect extending the DFS namespace to engulf additional namespaces. At a minimum Cairo ships with support for junction points to Novell's NetWare 4.x directory service.
- *RPC Name Service provider.* Cairo DS is a repository where RPC servers can register their existence so RPC clients can connect to them without knowing their address (e.g. by naming them or querying for them), or even without knowing their name (by querying for them).
- *X.500-like.* Cairo DS uses a conceptual model resembling the X.500 DS standard: a tree-structured namespace where every node is an object with a class and one or more properties.
- *Flexible administration.* The DS provides a single site of administration for many aspects of the Cairo network. Since DS uses flexible, ACL-based security, the network may have anything from a single administrator to many different administrators for many different things. And since the DS is uniformly accessible from all Cairo machines on the network, network administration can be done remotely from any Cairo machine on the network.

Benefits

(See also the benefits in the Distributed Filesystem section.)

Users can browse or query a single place to find resources of any type (e.g. documents, users, printers, disk volumes, groups, domains). All of the facilities provided by DFS (e.g. query and summary catalogs) can be used on all types of resource. This allows for significant simplification of the UI: e.g., finding a user to send email to is no different than finding a printer or a spreadsheet.

Administration is simplified on Cairo since the administrative UI (e.g. creating a user) is merged with the Cairo shell (which already supports things like creating a document), and administrative security (e.g. permission to delete a printer) is merged with DFS security (e.g. permission to delete a document).

Resources provided by foreign network operating systems are accessible as seamlessly as Cairo resources: the user need not know that the printer he is sending a job to is on a Novell server, or that the user he is sending an email to is a PROFS email user.

Services get a well-defined, standard place to store their configuration information: the file that describes them in the DS. And services no longer need to send out broadcasts to announce their existence, since clients can now find them by asking the DS. For example, the LANMan/WfW Browser and the RPC Locator are now obsolete (but are retained for downlevel compatibility).

Disconnected operation is supported by the ability to have a locally stored address book and the ability to have DS data describing local resources and accounts stored on the local workstation.

Cairo DS's conceptual similarity to X.500 eases the future creation of gateways between Cairo DS and other directory services.

Special-purpose system databases such as Security Accounts Manager (SAM) are no longer needed because the DS serves as a generic database for system resources.

Compatibility

Cairo DS does not use the API, wire protocol, or on-disk representation of any other manufacturer's directory service. Gateways are provided to allow Cairo DS to incorporate resources from Novell NetWare 3.11, Novell NetWare 4.0, and possibly Banyan Vines and DCE networks.

It does use the same API, wire protocol, and on-disk representation as the Cairo components it is based on: OFS, DFS, and catalogs.

Since the DS is used to find RPC servers, mechanisms are provided to make the Cairo DS interoperate with the NT 3.1 RPC Locator. Since the DS is used to find file/print servers, mechanisms are provided to make the Cairo DS incorporate information from the Windows for Workgroups NetBIOS Browser.

Non goal

The first Cairo release will not support non-Microsoft machines accessing the Cairo DS. (The protocols are unique, and no inbound gateways are scheduled to be created.) Workgroup Apps writers might want to do this gateway work in the Cairo timeframe, especially for Macintosh.

Under consideration

It's not clear what tools we will provide to assist in listing foreign resources in the DS, or how automated this process will be -- especially for foreign systems other than Novell.

It's not clear whether Cairo DS will be used as a name service by network transports (e.g. an IP address server).

Replication - Yuval Neeman, Aaron Contorer

Goals

This feature provides replication of OFS storage catalogs (i.e. sets of files on OFS disks) to increase availability, both for fault tolerance and for partially connected networks (WANs). It may also be used for load balancing. It is used by Cairo to replicate domain controllers, to create and maintain summary catalogs, and for disconnected operations.

Benchmark

Cairo replication will be a super set of the replication features in Lotus Notes 2.0.

Features

- *File replication* The Cairo replicator is used to replicate collections of files. These may be single-stream files, or complex multi-stream OFS files.
- *Security account replication.* The Cairo replicator is used to replicate user accounts, groups, profiles, etc., which are in turn implemented as OFS files.
- *ACL replication* Access control lists are replicated along with the files they affect. Thus, replicating a file does not affect anyone's security permissions on the file.
- *Efficiency.* Whenever practical, the replicator will copy only those files that have changed, rather than copying the entire storage catalog. OFS mechanisms such as update sequence numbers and deletion logs are used to efficiently determine which files have changed.
- *Configurability.* The customer has great control over the replication of any storage catalog: how many replicas to have (zero or more), how often to update, time of day and/or day of week for update, single vs. multiple masters, etc.
- *Weakly consistent replication.* The Cairo replicator uses weak consistency. This means that there is no guarantee that two instances of a storage catalog contain exactly the same information at any time. Weak consistency is more suitable for an environment in which machines are not always available. It has a drawback in that a user may access an object that is not up to date, but it allows files to be accessed even if some of the replicas are offline.
- *Single and Multiple master replication.* Replica sets may be configured to have a single writeable copy and multiple readable copies (used for data distribution) or multiple writeable copies (used for domain controller replication and for Notes-like textbases).
- *Conflict resolution.* If there are multiple writable copies and more than one copy is updated at the same time, a conflict may occur; the replicator invokes generic rules or class-specific code to reconcile such conflicts. Cairo provides a limited set of class-independent reconcilers (last write wins, Summary Catalog); these may be supplemented by class-specific reconcilers which take advantage of class semantics to do more sophisticated reconciliation (e.g. merge or generate warning).
- *Partial replicas.* A read-only replica may be configured to contain just a subset of the files contained on the master. This is useful for creating and maintaining summary catalogs, or for distributing subsets of a textbase.
- *Public and Private replicas.* Public replicas are created by an administrator. Each replica exhibits the same access and ownership as all of the others (i.e. ACLs are also replicated). Private replicas may be created by anyone, and incur a lower resource overhead because they are only sync'd with the other copies upon request, not automatically.
- *Name transparent replicas.* In concert with DFS, a set of replicas can be configured as a name-transparent set -- that is, all the copies will have exactly the same name; the existence of multiple copies is thus transparent to the caller. DFS will make volume selections based on the access requirements and some form of load balancing (i.e. solve the \\products1 & \\products2 hack).

- *Non-transparent replicas.* Alternatively, a set of replicas can be configured to each have a different name in the DFS. In this case, the DFS does not automatically select the best replica; it always selects exactly the replica corresponding to the pathname requested.
- *Replication over non-fully-connected networks* A replica set may be configured to include nodes that are not directly connected to each other, as long as there intermediate nodes that are connected to all nodes in the set. (In other words, *transitive* replication is supported.)

Benefits

High availability of system and user data even on networks that are not always connected. Can be used for: high availability of critical data, load balancing for heavily used servers, data (binaries) distribution, sharing of data across WANs, synchronization of data for mobile machines, and maintenance of summary catalogs.

Compatibility

Cairo replication is not compatible with the Replicator service (for replicating files) provided with the Windows NT Advanced Server.

Name transparent replicated volumes will not enforce locking and exclusion semantics across replicas, e.g. if a user opens an object in DenyWrite, other replicas still may be opened for write.

Non goal

Cairo replication will not provide strong consistency, nor will it provide absolute guarantees that a given copy is up-to-date within some time window.
Cairo replication will not use messaging as a transport, to allow sharing of data between disconnected networks (this is a Notes feature).

Cairo replication will not work on non-OFS disks.

Under consideration

There are some technical problems with ACL replication, e.g. NT capabilities cannot be replicated. Not clear what our fallback plan is if we can't work this out.

System Management - Dan Plastina

Cairo system management encompasses the many tasks that system administrators face in their daily duties. With the combined efforts of Microsoft and ISVs during our Beta program, we seek to provide broad spectrum of solutions for system management tasks by product release time. When complete, Cairo will offer solutions to many management problems such as lack of remote management support, software distribution and maintenance, asset data collection and monitoring support.

Goals

- ✓ Cairo should be easier to manage than the competition and its progenitors.
- ✓ Cairo is to provide the infrastructure and tools necessary to improve management of applications and systems participating in a Cairo environment.

Interoperability with industry management standards and other versions of Windows is the purpose of Hermes, Microsoft's enterprise management solution. For this reason, integration, or at a minimum, co-existence with a future version of Hermes is a must.

Benchmark

For the most part, system administrators do not expect ultimate performance from their tools -- this is not to say they don't expect the system itself to be responsive. In almost every case, administrators will trade flexibility and added functionality for speed. Some components have performance goals, they will be mentioned below.

We have few real competitors in the area of system management. More numerous and improved solutions simply increase the deployment of Cairo as this task is made simpler.

On the PC platform, Novell's Netware Management System (NMS), is a management solution that is geared towards network management with some Netware specific support. This solution has little effect on our offering as they chose not deal with the task of managing Windows.

For performance evaluation, the management tools included by other NOS suppliers is a good measure. In this category are Vines, Netware and our own LAN Manager. Other tools sold by ISVs should be compared when appropriate (e.g.: Inventory applications are many).

Features

Base Infrastructure

- *Distributed Event system*. Cairo offers a distributed, asynchronous event system for use by system management and other components of Cairo. The event system does not support queuing but does offer exemplary event message content and sink/source interaction. The event system has a default configuration that supports transparent, automatic registration of added resources. Note: This is a low bandwidth system -- around twenty five messages per second.
- *Configuration Profiles*: Profiles are configuration objects. This infrastucture permits functionality similar to that of WinLogin™ and NT user profiles across all participating applications. Profiles adapt well to disconnected operation as well as roaming users.
- *Inventory Management*: Hardware and software inventories are supported in Cairo. In fact, inventories can be built from any object (e.g.: Users and Printers). A number of pre-configured inventory catalogs are supplied and others can be constructed using the summary catalog building tools.
- *Job Control Language*: Batch-oriented administrative tasks can be handled with one of three end-user programming solutions: Object Basic, Visual Programming, and the Windows NT 3.1 batch language. Many Cairo components are instrumented such that they can be called by these languages to permit increased automation.
- *Annotation to Help files*: Cairo's help system permits administrators to append data to shrink-wrapped help files.
- *Scheduler*: Cairo offers a system service that permits registration for event notification based on an application specified schedule.

Software Management

- *Usage Model*: Locating and installing applications has been made much simpler. Gone are the days where a user needs to know which \\server\share an application is located on.
- *System Support*: Cairo will define a programming interface for software management. This interface will support registration of applications, klass associations, and other requirements associated with introducing and using applications in this new environment.
- *Cleaner Application Installation and Setup Toolkit*: Cairo uses a configuration data file called the Package Definition Format (PDF). This file contains mostly information required to install, deinstall and otherwise control the application life cycle in the Cairo environment. The PDF, in conjunction with software installation interface, form a complete solution for ISVs developing commercial application installers. For those who prefer simplicity over flexibility, a setup toolkit for Cairo will be provided upon product release. Hermes has agreed to use PDFs in their Windows NT 3.1 product.

- *Installation and Distribution*: Administrators can target a Cairo workstation for remote application installation. Hermes will support down level clients and asynchronous distribution.
- *Shared Usage*: Applications participating in the profile and software management infrastructures are much simpler to deploy on a server for shared use.
- *Setup Toolkit*: The Cairo team will not develop this. SWAT will be asked to develop this.

Storage System Management

- *File system Management*: A GUI disk management tool will perform all file-specific operations such as format. It is designed to be extensible by ISVs to support additional functionality.
- *Storage system maintenance (DFS Management)*: A DFS management tool will perform all DFS-specific operations including namespace and replication configuration and monitoring. Storage maintenance is fully supported on OFS volumes and with reduced functionality on NTFS, DOS and foreign volumes.
- *Backup*: Cairo will supply a simple backup/restore solution. The provided solution will be capable of dealing with Cairo objects and will be integrated with the profile architecture to provide intelligent backup of applications.
- *File Migration*: Cairo will supply a simple file migration solution. The provided solution will be capable of migrating Cairo objects to other locations in the distributed namespace. Support for file migration maintains our position with respect to Netware 4.0

Service Management

Cairo has many services each of which has particular management requirements. Each of these services are manageable via the service management infrastructure. Here is a list of the services that are known at this time:

Security (Kerberos)	Binder (COSMOS)	Replicator	Print
Software Distribution	Content Indexer	DFS	Network
Summary Catalog builder	Software Licensing	Event Manager	Mail
Global View	Shared Tape Drives		

Monitoring and Performance/Fault Management

- *Performance monitor*: The Windows NT 3.1 PerfMon will be ported to Cairo and extended to support new monitoring data that Cairo makes available.
- *Fault tolerance*: All fault tolerant features of NT will continue to exist in the Cairo environment. Enhancements, such as mirrors of striped sets and increase set sizes are made to better the level of support needed come 1994-5.
- *Error Log*: Windows NT 3.1's solution for error logging will be kept and incrementally improved.
- *Auditing*: NT's solution for auditing will be Cairo-ized Some effort will be invested into improving the usability and readability of the audit trail.
- *Help Desk Facility*: Delphi forms will be used to provide a simple means by which users can send service requests to MIS. The use of forms and object basic permit easy modification by the customer and support for automatically entered data.

Benefits

- *High level of integration*: System Management in Cairo has been designed into the product from the very start. This offers a greater level of integration and strong infrastructure upon which to base future additions to the produce -- may they be Microsoft or third party additions.
- *Drag and Drop*: Management by drag and drop is a strong part of Cairo system management. Whenever possible, there is a drag and drop solution to each task. This simplifies otherwise complex tasks (e.g.: Drag a user from domain A to B will move the user account from one domain to the next).

- *Centralized, Decentralized and Mixed Management Styles*: A multi-domain Cairo network can be managed centrally and decentralized management of the individual servers and workstations is performed at the domain level. Mixed management is automatically supported as central management does not preclude management at the domain level. Administrators can mold Cairo to their current management styles or choose one that was not possible with their previous system.
- *Extensibility*: Management tools are integrated with the Cairo shell programming environment to provide the administrator with the flexibility needed to truly alter the environment to suit their particular needs. This level of flexibility currently only exists to those capable of coding in a professional development language like "C".
- *Solution to common problems*: Cairo system management has solved many of the larger problems that IS shops face today. Cairo offers software management to reduce costs of upgrades and deployment of new applications. Cairo offers an enhanced configuration profile model to facilitate networking of Windows and hosted applications. Cairo includes a comprehensive, flexible solution to asset management. Remote configuration is supported in Cairo thus reducing the number of on-site visits. Knowing well that Microsoft can't solve all problems, an infrastructure to assist in creation and development of management applications is part of Cairo. All these efforts together formulate a clear effort to ensure that Cairo is manageable.

Compatibility

Chicago

No special support exists in Chicago v1.0. Chicago v1.1 could be made a better Cairo client to strengthen our system strategy. A second, revised version Chicago released after Cairo could address this issue.

NT/Server and LAN Manager 2.x

Cairo will support remote management of Windows NT 3.1 user accounts and servers to the extent that NT supports remote management today. Support for LAN Manager 2.x is secondary and will be done via Windows NT 3.1's mapping layer. Any shortcomings introduced by taking this route shall remain.

Hermes

The Hermes team has agreed to perform the work required to integrate or permit coexistence of Hermes within Cairo. This work is considered Hermes - Product 2 which will be a Cairo based application. In addition to the integration work, *Hermes for Cairo* will enhance enterprise management and integration for the Cairo platform.

NetView & SNMP

Netview is an industry standard management console and SNMP is an industry standard management protocol. Cairo will offer support based on the efforts of the Windows NT 3.1 group. See below section on this topic.

Non goals

- *Network Management and Planning*: Cairo is not directed towards supporting network management, or stated differently, management of the network fabric itself – bridges, routers and segments. Network planning will be supported in document form only. ISVs will be asked to fill out this area.
- *Accounting*: Not supported in Windows NT 3.1 – Not supported in Cairo.

Integration with *Hermes For Cairo*

- *Management from Enterprise Consoles – NetView and SNMP*. Cairo will neither augment nor break remote monitoring and control from a NetView™ console or from SNMP. The extent of the support is based on the Windows NT 3.1 offering and is being developed by the Hermes team. Due to weak security in the current definition of SNMP, remote configuration of Cairo via SNMP will not be supported.

- *Installation and Distribution*: No forms of asynchronous distribution is supported in Cairo without Hermes. Support for Macintosh and other clients is also not planned at this time.
- *Licensing*: Licensing will be a Microsoft supported service by the time Cairo is released. Hermes will introduce it in Q393. Her Only minor integration work should remain to be performed.
- *Remote Console*: Cairo is not directed towards using remote console as we support remote management without taking over remote keyboard, console and mouse. Hermes will provide this support.

Under consideration

- Nothing "on the fence" at this time.

Security and Authentication System - Pradyumna Misra

Cairo provides a secure distributed application environment. Cairo services rely on the underlying operating system to provide local security. For the local security authority of the operating system to perform access validation it needs to identify and authenticate the user and then acquire the granted authorization information. It is the role of the *distributed security system* to provide the local operating system with this information about a user in a networked and distributed environment. The distributed security system allows client applications and services to ensure the integrity and the confidentiality of the data that they exchange. Cairo distributed security augments local security mechanisms provided by NT and unifies them with Kerberos authentication.

Goals

Cairo provides a superset of the already rich distributed security functionality provided by Windows NT Advanced Server. However, it does this in ways consistent with reducing the burden on administrators, increasing manageability, making security features more accessible to application writers, and simplifying the use of a Windows client in a heterogeneous network. Besides single enterprise-wide logon and usable security features, the Cairo security model is scalable to all possible configurations including Standalone, Domains, and Enterprises consisting of multiple domains.

Benchmark

OSF's DCE 1.0 provides most of this functionality - authentication and authorization, integrity and privacy, authenticated RPC but does not provide digital signatures or transfer of authorization. Novell's NetWare 4.0 provides a subset of this functionality - authentication and authorization services only. Lotus Notes is not a direct competitor for this functionality but provides digital signature and privacy services.

Features

- *Kerberos Authentication.* The authentication component manages logon of a user in the distributed environment. Cairo uses Kerberos (version 5) authentication protocol developed at MIT and widely used. However Cairo implementation of Kerberos is RPC based unlike MIT's but similar to DCE's.
- *Single Enterprise-wide Logon.* A user needs to logon at most once during a Windows Cairo session. This logon is a Kerberos logon for every system in a Cairo domain. The exception to single logon is if the user tries to access resources provided by non-Cairo servers for which no credentials are available to the system.
- *Integrated Authorization.* The authorization component includes the validation of a user request and the registration of validated security tokens with the local security authority at the server node. The authorization component for Cairo takes advantage of certain advanced features in the Kerberos V5 protocol to implement a ***privilege attribute certificate (PAC)*** based authorization.
- *Integrity and Privacy Services.* Cairo provides per-message integrity and privacy services on each connection. The integrity service guarantees that data has not been tampered with during transit, and the privacy service

(AKA ***encryption***) ensures the confidentiality of the contents of the message. In addition to these services the applications can also request ***digital signatures*** which prove the identity of the user who created a message.

- *Authenticated RPC.* Security services have been integrated into Microsoft RPC and are accessible via a number of high level RPC runtime APIs that hide the complexity and nature of underlying security system. Applications can thus take advantage of commonly used Cairo security services by simply specifying desired level of security on binding handles. These level range from no security or default security to maximum security that includes integrity and privacy on each message exchanged.
- *Generic Security Service APIs.* A variation of Generic Security Service (GSS) API[3] will be used as the API layer for the security services. This allows security services to be available to RPC, redirector and server and applications without exposing the internals of underlying security system.
- *Interoperability.* Cairo security is designed to support multiple authentication protocols via security mechanism specific implementations accessible by generic security APIs. This permits addition of new authentication protocols for interoperability with non-Cairo systems. The primary authentication protocol is Kerberos that allows for easier interoperability with systems using Kerberos security.
- *Credential Management.* This features allows users to make connections to servers they normally use without retyping their passwords or other credentials. The system, with permission from the user, caches credentials into user's record for reuse across boots and makes them available to client upon successful logon.
- *Security Administration.* Unlike most systems that implement security database as a separate system Cairo security database is part of the directory service. It contains user names and passwords and the authorization information about user accounts. The security objects are implemented using the same paradigms and infrastructure as rest of the system components and can be manipulated or viewed using same tools thus reducing the learning curve.
- *Support for Large Domains.* Cairo supports large domains that generally mirror the organizational unit versus the departmental unit. Cairo domains are designed to scale from small to very large with minimal administrative burden. They will typically hold tens of thousands of users and computers.
- *Access Control Lists , Auditing, and Privileges.* The access lists and privileges control access to resources made available by Cairo servers much like Windows NT 3.1. The format and contents of ACLs are a superset of those in Windows NT 3.1 and are compatible. Cairo also provides same level of auditing support as in NT 3.1.
- *Transfer of Authorization* Cairo provides a ***proxy*** mechanism that allows you to authorize a remote service to act on your behalf, but retain some degree of control of actions that remote service performs on your behalf. (This is also known as ***delegation***.)
- *Account Lockout.* Cairo supports account lockout mechanism to prevent dictionary attacks and it does so without causing denial of service at the same time.

Benefits

Security becomes very important in a distributed environment since unlike non distributed operating systems, there is no one single entity that can be trusted to authenticate users and programs, to protect the authenticity of messages exchanged between them, and guard against unauthorized resource access.

Cairo's distributed security system augments the local security mechanisms provided in Windows NT 3.1 by adding Kerberos authentication and provision for inter-operating with non-Cairo security systems such as DCE's and thus preserving the investment in existing systems. Cairo security offers many advanced services beyond Windows NT 3.1 including integrity and privacy services, mutual authentication, digital signatures, proxies, and scalability to any level of hierarchy that provides for ease of administration and resizing.

Most of Cairo security activities happen without the knowledge of users and seldom get in their way after the successful initial logon and that includes such complex things as proxies and integrity protected message exchanges. Integration of security services with RPC runtime simplifies the programmers' perspective of security

[3] GSS API has been defined by J. Linn of Digital Equipment Corporation.

services and allows applications to use most of Cairo security services without having to learn the details of underlying security system.

Compatibility

Cairo will provide a superset of the network operating system interoperability provided by Windows NT 3.1. Specifically Cairo clients will be able to access resources on servers running Windows for Workgroups, Windows NT Advanced Server, LAN Manager 2.1 products. Cairo servers will allow access to their resources to clients running Microsoft networking software.

It will be possible for administrators to manage NT 3.1 domain controllers and workstations from Cairo machines. The Cairo DCs will also support NT 3.1 style pass-thru authentication to enable a gradual migration of NT 3.1 domains to Cairo domains.

In addition to the above Cairo will provide mechanisms that allow inter-operation with DCE conformant systems and popular network operating systems like NetWare and Vines.

Non goals

We explicitly decided not to pursue the following:

- Use public keys for authentication.
- Provide comprehensive protection against viruses and worms.

Under consideration

- Digital signature formats - should we use an industry wide standard ?
- Security inter-operation between organizations.
- It is not understood which pieces of software (redirectors, stacks etc.) will be developed by Microsoft and which will be done by third parties, towards achieving our inter-operability goals.
- Is there a requirement to co-exist with DCE systems on the same local network (domain) i.e. allow Cairo KDC to issue tickets for DCE systems and vice-versa?
- Is inter-operability with systems running MIT version of Kerberos desired ?

Network Interoperability - Aaron Contorer

Cairo is an open system, designed for interoperability with and seamless use of, heterogeneous network services.

Goals

Cairo clients have access to network resources provided by Cairo, Microsoft down-level, and foreign operating systems. Differences between these providers are hidden from the caller and end user where feasible, though full provider-specific functionality is still available when desired. Also, resources on Cairo servers can be accessed by downlevel Microsoft clients, though these clients may see no added value in a Cairo server compared to a Windows NT 3.1 server.

Cairo supports interoperability with the following systems: Windows NT, Windows NT Advanced Server, Windows for Workgroups, LAN Manager 2.x, and Novell NetWare.

Support for interoperability with OSF DCE compliant systems will be pursued through third-party DCE technology providers. Cairo's RPC and Kerberos security are interoperable with DCE without the need for third-party add-ons.

Benchmark

Windows for Workgroups provides concurrent access to resources provided by multiple network systems, but the user is relatively aware of the source of the various resources; access is not seamless.

MS Mail with gateways and directory exchange services provides the ability to enumerate and send mail to users on a variety of back-end mail platforms in a seamless way. This is a good representation of what Cairo aspires to do for files, printers, and other network resources as well.

Features

Cairo interoperates with down-level and foreign systems such as Windows NT, Windows NT Advanced Server, Windows for Workgroups, LAN Manager 2.x, and Novell NetWare systems in the following functional areas:

- *Fileserver access.* Cairo clients have transparent access to fileservers on foreign systems. Cairo supports the installation and simultaneous use of multiple file redirectors, and the DFS makes selection of a server volume transparent, even across multiple server operating systems.
- *Printer access.* Cairo clients have transparent access to printers on foreign systems as in Windows NT 3.1, and these printers can now be conveniently selected from the Cairo DS.
- *Publication of foreign resources.* Foreign resources can be listed in the Cairo Directory Service and in catalogs. Once listed, these resources (i.e., print queues, fileservers, and disk shares) can be found and used in the same manner as Cairo resources.
- *Foreign directory service incorporation.* The Cairo Directory Service allows the creation of *junction points* to incorporate other manufacturers' entire directory services as part of the DFS namespace – in effect extending the Cairo DFS namespace to engulf additional namespaces. This lets Cairo clients browse through the foreign directory service as if it were simply a part of Cairo's DFS. Cairo provides junction points and/or catalogs that incorporate Novell NetWare resources automatically, and ISVs may add support to incorporate additional foreign directory services.
- *Email support.* The Cairo Global Catalog can contain users from foreign systems and their associated email addresses, allowing both the discovery of email addresses as well as the actual delivery of mail to, and from, non-Cairo email users.
- *Security.* Cairo clients can seamlessly access servers using any number different security systems, by installing security provider software on the client. As servers, Cairo machines support access using Cairo protocols as well as downlevel Microsoft protocols (e.g. downlevel SMB using challenge/response authentication).
- *Remote Administration.* An administrator sitting at a Cairo machine can administer both Cairo machines and Windows NT 3.1 machines using Cairo-style tools.
- *Browser and Locator support.* Cairo supports the NetBIOS Browser and the RPC Locator: outbound, to enable discovering Cairo servers from downlevel Microsoft clients; and inbound, to let Cairo clients discover downlevel Microsoft servers. (Natively, Cairo uses the DS to get more functionality than the Browser provides).
- *Downlevel SMB service.* Cairo supports downlevel versions of SMB to allow downlevel Microsoft clients to use a Cairo machine's file and print services, and to use these services on downlevel servers.
- *Remote Procedure Call.* Cairo provides support for Microsoft Remote Procedure Call (RPC). Cairo's RPC is compatible with RPC on all downlevel Microsoft systems including Windows NT, Windows for Workgroups, Chicago, Windows 3.1, LAN Manager 2.1, DOS, LM Unix, and the MS Macintosh client for Windows NT. It is also interoperable with OSF DCE RPC, which allows Cairo clients to access DCE compliant servers.

Benefits

Resources provided by foreign and down-level network operating systems are uniformly accessible from Cairo clients. (i.e., The user need not know that the printer he is sending a job to is on a Novell server, or that the file he

is accessing is on a Windows NT server.). Additionally, non-Cairo clients are allowed access to native Cairo network resources. This allows an installed base of heterogeneous client and server systems to interoperate effectively within the Cairo enterprise network.

Compatibility

Cairo will provide a superset of the network operating system interoperability provided by Windows NT, Windows for Workgroups, and LAN Manager 2.1 products. This includes complete compatibility in the following areas:

- *MS Remote Procedure Call (RPC).* Cairo uses the same RPC for interprocess communication that is used in Windows NT, Windows for Workgroups, Chicago, Windows 3.1, LAN Manager 3.2, DOS, LM Unix, and the MS Macintosh client for Windows NT.
- *Transports* Cairo will support network communication through existing protocols on downlevel MS systems, including Named Pipes, NetBIOS, WinSock, TCP/IP, DECNet, SPX, and downlevel SMBs.

Non goals

Cairo will not provide interoperability support for *all* foreign network operating systems. Instead, we will provide documentation and sample code that will enable third-party vendors to develop the software necessary to interoperate with Cairo systems.

Cairo does not provide support for the enumeration or access of Cairo resources from non-Microsoft clients.

Cairo does not provide support for administering any machines except Cairo and Windows NT 3.x machines.

Under consideration

Support for interoperation with additional network operating systems. (e.g., Banyan, Apple, IBM, etc.)

Degree of support for DCE will depend on market demands and competitive strategy. At a minimum Kerberos and RPC will interoperate, but directory service and DFS interop are not yet planned.

Network Configurations - Aaron Contorer

Goals

Cairo works in a variety of network configurations, all the way from standalone (no network) to large networks, providing transparent access to all available resources in any network configuration.

Benchmark

Macintosh and Windows for Workgroups provide a smooth transition between standalone and peer networking, and Windows for Workgroups also allows for a moderately smooth transition from peer to client-server networking with LAN Manager or Novell. Windows NT 3.1 can be configured in either a standalone/peer server role, a domain member role, or a Server/Domain Controller role, though the transitions between some of these roles could be simplified.

Features

- *Standalone* A Cairo machine can be configured to be completely self-sufficient, never attempting to contact another computer.
- *Domain.* Two or more Cairo machines may be networked together to form a cooperating system called a *domain*, consisting of one or more domain controllers (DC's) (aka Advanced Servers) plus one or more clients (aka Workstations).

  - *Network Client.* A Cairo machine can be configured to act as a network client of one or several types of server, even on a network where none of the remote machines runs a Microsoft operating system. All the network resources can be seen by the Cairo machine in a client-side subset of the Cairo directory service. A Cairo client can simultaneously act as a server of disk volumes, printers, serial devices, tape drives, email, RPC, mailslots, and named pipes.
  - *Domain controller* A Cairo server may be configured as a DC to act as a centralized point of system configuration and administration. Every Cairo network has at least one domain controller. The domain controller (aka the Advanced Server) has all the capabilities of the client, plus it is a directory service server, a security server, and a DFS name resolution server, and may take a special role in running downlevel support services such as RPC Locator and NetBIOS Browser.
- *Mixed domain.* A domain run by a Cairo DC may also include Windows NT 3.1 machines. All of the DC's in the domain must be Cairo DC's.
- *Multiple primary domain controllers.* A given domain may have more than one machine acting as domain controller. In this case all such machines are equal: they all accept writes and they keep each other approximately up to date using the Cairo replication system.
- *Multi-domain network.* A large set of Cairo machines may be configured as more than one domain, with each domain trusting the other domains to log on users. This lets the customer put the accounts in any given domain(s), allowing multiple zones of administration or central administration, even across an intermittently connected wide-area network. Unlike Windows NT 3.1, Cairo does not require the administrator to create explicit "trust relationships" between domains: an account in any domain is recognized in all domains throughout the organization.
  - *Mixture of NT 3.1 domains and Cairo domains.* Above we noted that a domain run by a Cairo DC may also include Windows NT 3.1 machines. In addition, a company may have other domains run by Windows NT 3.1 domain controllers. (Such machines cannot have Cairo machines as members.) To unify account management across the domains, the Cairo domains support being trusted by the Windows NT domains there is no need to duplicate accounts.
- *Disconnected operation.* Any Cairo machine can temporarily remove itself from the network at the user's request, and will then continue to operate correctly and securely even after being unplugged from the network connection. Before leaving the network, the user may indicate a desire to take with him some information from the network (e.g. some documents or an address book); these are then automatically stored (cached, in effect) on the workstation's hard disk and can be automatically returned to the network after the machine is reconnected. (See "Replication" section for more on this.)
- *Dialin.* A Cairo machine may act as a full participant in a network over a dialup line.
- *Intermittent connectivity.* Different domain controllers, whether in the same domain or different domains, can function properly and share data with each other even in an environment where the network link between them is up intermittently. When the link is down, each of the separate "islands" continues to work.
- *Avoids unnecessary connectivity restrictions.* Two machines may both be part of the same Cairo network even if they have no connectivity to one another. Only domain controllers and machines offering services have to have connectivity to the member machines of other domains; member machines that use only resources in their own domain, and that wish to provide no services nor performance data to machines in other domains, can operate correctly without connectivity to any other domain.
- *Mixed CPU types supported.* In a Cairo network, no machine depends on any other machine to have a given type of CPU. For example, a single domain may have one Intel-based domain controller, one Alpha-based domain controller, and a bunch of MIPS-based members.

Benefits

Because of its network scalability, Cairo is useful to customers regardless of the size or configuration of the group in which they work: anything from a reclusive laptop user in the woods to a giant multinational corporation. It continues to work in more or less the same fashion even as a company grows or shrinks, or even as a user comes and goes from the office.

The significant flexibility in configuring domains and domain controllers allows customers to choose a wide range of administrative styles, and to compensate for limited connectivity through replication or multiple domains. Support for mixed CPU types gives the customer broad flexibility about what types of machine to buy.

Compatibility

(See also the above section on Network Interoperability.)

Cairo's machine roles resemble the machine roles that Windows NT 3.1 uses. NT has standalone, peer server, Server (backup domain controller), and one primary Domain Controller per domain, while Cairo has standalone, workstation, and multiple Domain Controllers per domain. A Cairo domain may include Windows NT 3.1 machines and can perform authentication for them using Windows NT 3.1 protocols.

Limitations

While Cairo is quite flexible, Cairo network configurations still place some restrictions on how the customer may configure the physical network. In particular:

- Every Cairo machine that is a member of a domain must have connectivity to its domain controller. (If the DC is replicated, connectivity to at least one replica is sufficient.) If the member is starting up and cannot communicate with the DC, it will still work locally but certain functions will be temporarily disabled: e.g., the member cannot accept connections from remote clients and cannot participate in replication. This is what we call *disconnected operation*.
- Due to how Kerberos works, a Cairo client setting up a connection with a Cairo server in another domain must have connectivity to that server *and* a DC of the client's domain *and* a DC of the server's domain.

Non goal

Cairo does not support a pure peer network; at least one of the Cairo machines on a Cairo network must be a domain controller.

Cairo does not support *seamless* disconnected operation; instead the user who plans to disconnect the machine from the net must indicate what information he intends to take with him, and he will be quite aware when he is off the network because many resources (e.g. remote printers) will be unavailable.

Cairo does not support *mobile* operation, wherein a mobile machine continuously comes in and out of contact with the network due to limited radio range and continuously adapts to these changing conditions. Not all Cairo components will deal gracefully with this sort of unpredictable intermittent connectivity – e.g. print requests will not be silently queued locally until connectivity returns.

Although Kerberos technology in theory allows it, we will not do the work to make a Cairo domain hold accounts for non-Cairo Kerberos clients, e.g. DCE machines running the OSF/1 Unix operating system.

Mail Server - Aaron Contorer

Goals

The Cairo mail server is a general-purpose, industrial-strength X.400 messaging server integrated with the Cairo system. It is the upgrade to Microsoft's Enterprise Message Server product.

Benchmark

Messages per second throughput will be faster than any store-and-forward email service Microsoft has ever shipped prior to Cairo – faster even than the Enterprise Message Server that ships before Cairo, due to structural optimizations in the code and the use of OFS. Functionality will be a superset of any email service Microsoft has

ever shipped prior to Cairo, and will be at least comparable to the functionality of X.400 mail servers provided by other companies.

Features

Cairo Integration Features

- *Message store integrated with OFS* Users' messages and message folders can be stored on the server, and are implemented as files and directories on OFS. There is no separate database code or format: OFS is the store.
- *Mail accounts integrated with user accounts.* The mail server does not require separate email accounts to be created for Cairo users. A Cairo user's logon account is his email account as well -- stored and administered as a single object.
- *Security groups integrated with mail groups.* The mail server treats security groups as a legitimate target for email, and it will expand them in the same way that other mail groups (aka "mail aliases") are expanded.
- *Integrated with Cairo directory service.* The mail server uses the Cairo directory service as the distributed database to store and retrieve mail information, such as: users' mail addresses; user preferences; routing information; distributed system configuration.

Other Features

- *The features of MIR and EMS.* The mail server will provide the features of Mercury Internal Release (the mail server scheduled for Microsoft-internal release in 1993) and of the Enterprise Messaging Server (which supersets MIR and is scheduled for release in the first half of 1994). For clarity, a few of these features are redundantly included in this list.
- *True client/server architecture.* The mail server defines a clean RPC interface between the client machine and the server machine, rather than using file-sharing or other non-transparent communications.
- *High throughput.* On a single-processor 486/33, the mail server can accept and deliver messages with a sustained throughput of at least 2 messages per second for the case where the receiver's mailbox is not remote from the server.
- *Low latency* Messages which are tagged as high-urgency will be fully processed and delivered within at most a few seconds of the client's request to send them, assuming the server is not given a large number of high-urgency messages at the same time.
- *Support for gateways.* Gateway software can be installed on the machine running the mail server, allowing interchange of mail with non-X.400, non-Microsoft mail systems.
- *Support for downlevel (file-sharing-based) MS Mail clients.* A server-based gateway is included to allow downlevel MS Mail clients to use the Cairo mail server.
- *MAPI.* The mail server's client side exports the WOSA Mail Application Programming Interface (MAPI 1.x) as its primary usage API.
- *Flexible, distributed architecture.* Multiple Cairo mail servers can be installed within a single organization and will interchange messages automatically. Organizations with huge domains can install multiple mail servers in a single domain to increase capacity.
- *Clean, easy installation.* Mail server installation is integrated with Cairo installation, imposing very little mandatory additional work on the administrator running Setup.
- *Upgrade from EMS.* The mail server is compatible with EMS, and install software is provided to upgrade an EMS server to a Cairo mail server.

Benefits

The Cairo mail server provides high-performance, high-reliability, richly functional email fully integrated with the Cairo system. Integration with Cairo DS, accounts, and groups ensures that email does not create a separate management burden. Integration with OFS means that all of OFS's capabilities apply to email messages -- e.g.,

space-efficient storage; direct access to mail messages by file system APIs; access control lists (ACLs); content indexing and query.

Compatibility

The mail server can fully interoperate with other X.400 mail servers, including those produced by other manufacturers, and with MS Mail 3.x post office servers.

The mail server includes any code necessary to fully support Chicago clients, who are expected to use a simple file-sharing protocol to send and receive mail.

The client side of the Cairo mail server is fully MAPI compliant.

Non goals

The mail server does not need to run on low-powered desktop or laptop workstations; it may use hefty amounts of disk and RAM if this improves time-to-market, performance or functionality.

The mail server does not need to conform to industry standard messaging protocols when talking to other Microsoft-produced mail servers; it may switch to non-standard protocols where available if there is benefit in doing so

Under consideration

Does the mail server have to run on a machine that is not a domain controller? (There are optimizations possible if the answer is "no".)

Remote Procedure Call - Deborah Black

Goals

Microsoft Remote Procedure Call (RPC) is a transport-independent interprocess communication mechanism, which simplifies the development and execution of client/server or distributed applications. In Cairo, RPC consists of MS RPC as released in Windows NT 3.1, with enhancements to take full advantage of the Cairo Directory Service and Cairo Distributed Security. Applications written with the Cairo version of MS RPC are completely interoperable with all existing MS RPC applications, and therefore interoperable with OSF DCE RPC applications.

Benchmark

MS RPC is currently supported in Windows NT, Windows for Workgroups, and Windows Chicago. In Cairo, RPC will provide a superset of existing RPC functionality and will maintain or improve performance in the key areas of size and speed.

Open Systems Foundation's Distributed Computing Environment (OSF DCE) specifies support for RPC. Several vendors, including DEC and HP, have DCE products in the marketplace. Cairo RPC applications interoperate with DCE RPC applications. (ie., A Cairo client may access a DCE server, and a DCE client may access a Cairo server.) However, the MIDL compiler does not support source code compatibility for DCE IDL files.

Other vendors such as Netwise and SUN Microsystems distribute RPC products which use a network data representation format (XDR) which differs from the NDR format used by MS RPC and DCE RPC. It is not our intent to interoperate with these products.

Features

- *Network transports supported.* In Cairo, MS RPC provides support for the following connection-oriented network transport protocols: named pipes, NetBIOS, TCP/IP and SPX. The following connectionless protocols are also supported: datagram NetBIOS, UDP, and IPX.

- *Authenticated RPC.* Security services have been integrated into Microsoft RPC and are accessible via a number of high level RPC runtime APIs which hide the complexity and nature of the underlying security system. RPC applications can thus take advantage of Cairo network security services by simply specifying the desired level of security on binding handles. These levels range from no security or default security to maximum security that ensures integrity and privacy on each message exchanged.
- *Name service.* The Cairo RPC runtime supports the use of the Cairo Directory Service as a name service provider. This enables RPC servers to publish their services and RPC clients to locate servers of interest using the Cairo Directory Service.
- *Microsoft Interface Definition Language (MIDL) Compiler.* The MIDL compiler is a tool used to facilitate the development of distributed applications which use RPC. In Cairo, MIDL has been extended to support new constructs necessary for remoting Cairo and CairOLE interfaces, including the ability to pass interface pointers as parameters.
- *Interpreter.* A developer who wants to minimize the size of his RPC application may choose to use the RPC interpreter as an alternative to compiled stubs. The interpreter minimizes the duplication of marshalling and serialization code which is present in each RPC application's compiled stubs, by maintaining a single copy of this code in a .dll on the node. When using the interpreter, the application's compiled stubs contain a compact opcode representing the data type of each parameter. These opcodes are used by the interpreter at runtime to determine how the parameter should be marshalled.

Benefits

The MS RPC development tools allow developers to build distributed applications without handling the low level details of network communications programming. RPC applications may be easily ported to any MS network operating system, and will run over any of the network transport protocols listed above.

Compatibility

Cairo's MS RPC is compatible with MS RPC on all downlevel Microsoft systems including Windows NT, Windows for Workgroups, Windows Chicago, Windows 3.1, LAN Manager 2.1, LM Unix, and the MS Macintosh client for Windows NT. RPC applications written for these downlevel systems will execute correctly on Cairo. Additionally, MS RPC is interoperable with OSF DCE RPC, which enables Cario clients to access DCE compliant servers and DCE clients to access Cairo servers.

Downlevel RPC servers registered with the RPC Locator will be accessible to Cairo clients through the Cairo Directory Service. Downlevel RPC clients will have access to servers registered in the Cairo DS through the existing MS RPC name service interface.

Non goals

- Source code compatibility with DCE RPC applications.
- Interoperability with ONC RPC applications.

VI. Requirements: Developer Support

Cairo Development Environment/SDK - Janine A. Harrison

The SDK is what developers need in order to produce components, applications and services for Cairo. It contains the operating system, tools, samples, and documentation.

Goals

Clearly explain the Cairo programming model: Writing a Cairo application, or updating an existing application to be Cairo-aware, should be as easy as possible.

Provide re-usable code modules/components: The SDK will contain a programming environment (called the *construction site*) that will include online code samples (in both C and C++) for most of the standard programming tasks an ISV will want to incorporate. The SDK will also contain source code for all common Cairo components.

Provide programming tools which are replaceable: The construction site will contain a large number of tools to help the ISV complete programming tasks. These tools are replaceable or can be combined with those from other vendors

Become the Development Platform of Choice for Windows: The Cairo SDK will provide a complete development environment for creating and updating Windows applications

Simultaneous shipment of applications: The SDK team will provide the best possible environment and support possible to third-party vendors to ensure that as many Cairo applications as possible are ready to ship simultaneously with Cairo:

Benchmark

The SDK team will evaluate development kits from the following vendors:

- Apple
- SUN
- NeXT

Features

- *Cairo* The Cairo Operating System
- *Tools.* Development tools required to create applications for Cairo. These tools fall into two categories:

  Object-based Tools.

  Construction Site: A graphical programming environment the ISVs can use to create and update applications.

  Class Generator Reads .CDL files and automatically generates Cairo class code.

  TypeLib Generator: Creates type libraries (that include public interfaces, variables, and so on) used in implementations.

  Template Generator: Creates skeleton .CDL and .TDL templates and generates UUIDs that are used by ISVs to create customized class and interface descriptions.

  Browser: Used by ISVs to browse classes and objects.

  See also *Component Forms, the Cairo Design Environment, and Component Object Runtime.*

  Language-based Tools. Language level tools are still necessary in the SDK, especially for backwards compatibility. The following is the current list of language-level tools that will also be included in the SDK:

  Programmer's Editor

  Resource Editor

  Debugger

  Linker

  Language Compilers

  RPC Compiler

Resource Compiler
Resource Localization Tool
Source Profiler
Static Analyzers
Runtime Analyzers/Monitors
Help Authoring Tools (See *User Assistance*)
Setup (Installation) Authoring Tools

- *Code Samples.* The SDK contains source code samples (in both C and C++) that provide generic examples of how an ISV should take advantage of Cairo features. Current plans include the following samples:
  Generic
  MyClass
  MyInterface
  Property Set
  Property Sheet
  Context Menu
  Drag-Drop
  Docfile
  OFS
  Forms
  Component
  Connector
  Moniker
  Commands
  Content Index Filters
  RPC

- *Libraries.* Cairo libraries and include files.
- *Programming Documentation.* See *SDK Programming Documentation.*

Benefits

- *Consistency:* The SDK provides an environment of documentation, code samples, and programming tools that encourages ISVs to create applications that leverage the Cairo feature set and user interface paradigm.
- *Simple Migration:* The SDK will provide the information and tools the ISVs need to quickly and efficiently migrate their existing applications to the Cairo environment.
- *Extensibility:* The SDK construction site and component-based environment encourages ISVs to provide customized tools that enrich the Cairo development environment and customized components that extend the Cairo feature set.

Compatibility

Applications written using pre-existing Windows SDK tools can be upgraded using the Cairo SDK programming tools. In addition, ISVs will be able to use the Cairo development environment to create applications that run on pre-existing Windows and Windows NT platforms.

Non goal

The Cairo SDK will NOT plan to:

- Provide the best language development tools.
- Provide cross-platform development.

Under consideration

We are investigating other environments and third party tools to see if other vendors would be interested in porting their existing tools to our environment and (possibly) shipped with the Cairo SDK.

SDK Documentation - Karen Brown

Documentation for software developers will support both novice and experienced Windows programmers by providing extensive programming guidelines and both simple and complex code samples. The Cairo SDK documentation set includes both paper and online components. The paper documents contain in-depth discussions of the integral parts of Cairo. The online documentation focuses on quick reference information and reusable code samples that will enable programmers to code Cairo applications efficiently. For detailed information regarding SDK documentation objectives, strategies, and components, see the *Cairo SDK Documentation Plan*.

Goals

The goals of the Cairo SDK documentation set are to

- Make it easy for programmers to develop applications for Cairo
- Thoroughly explain Cairo architecture to programmers
- Provide concrete guidelines for upgrading an application to Cairo or creating a new Cairo applicaiton
- Meet the documentation needs of experienced Windows programmers
- Help programmers find answers quickly
- Make documentation available in the Cairo Advanced Development Environment
- Alleviate past criticisms of the Windows SDK documentation set.

Programmer Documentation

The Cairo SDK documentation set will support both novice and experienced Windows programmers by providing extensive programming guidelines and both simple and complex code samples. The online SDK will be a desktop running in the Cairo environment. Programmers will be able to quickly access online reference information and reusable code samples for Cairo system components.

The paper documentation for the Cairo SDK consists of the following books:

- ***Getting Started.*** Contains "what's new" and installation information
- ***Technical Overview.*** Contains conceptual discussions of object-oriented programming and the Cairo system architecture. Answers the following questions: "How to I upgrade my application to look and function like a Cairo application?" and "How do I create a new Cairo application?".
- ***Guide to Programming.*** Contains programming advice for experienced Windows programmers. Answers questions such as "How do I manage memory?" and "How do I implement a context menu?"
- ***User Interface Programming Guidelines.*** Contains specifications for writing Cairo applications that present a consistent user interface to the end user. Answers the question, "What user interface design and coding standards do I use to ensure that my application looks and feels like a Cairo application?"
- ***Programmer's Reference.*** Contains definitions of the Cairo classes, interfaces, methods, and APIs.

- ***Cairo Development Environment.*** Contains a description of the SDK desktop and instructions for how to use the programming tools
- ***Comprehensive Index.*** Contains index and glossary information for the entire paper documentation set.

We will test a preliminary version of the SDK documentation set for usability as soon as possible, so that we can incorporate feedback on the organization and content into the Beta and final versions of the documentation.

Benefits

Cairo User Education anticipates that this approach will provide these benefits:

- Provide better access to relevant information and sample code by putting all reference material and samples online
- Allow programmers to create code quickly by tying online information and sample code to component information in the development environment.
- Ensure consistent applications by providing the *Programming Interface Guidelines* with the initial SDK.
- Allow immediate access to information via the *Comprehensive Index.*
- Give different types of developers better tools by supplying a range of simple and complex code samples.

Non-Goal

We are not attempting to provide a comprehensive guide to programming with in-depth samples, shortcuts, tips, and hints. This will remain the province of third-party authors.

Device Driver Kit (DDK) - Janine A. Harrison

The Device Driver Kit (DDK) provides the independent hardware vendor (IHV) the tools and information they need to produce and/or update device drivers for Cairo.

Goals

Driver coverage for the vast majority of installed base of hardware for target marketplace. For example, LAN, video, storage, etc. for 386DX, 486DX, Pentium, MIPS, Alpha based upon projected numbers from DataQuest, IDC, MS sales figures, etc.

Benchmark

The desire is to improve upon the existing DDKs.

Features

- ***Plug and Play.*** Fold in the Chicago Setup/Configuration Plug 'n Play for system and device configuration. Important area to address is a better spec than ARC for system configuration of RISC and/or MP machines as Chicago doesn't address these.

General devices:

- Compression drivers.
- Complete built-in removable media support.
- Install to any storage device like CDROM, tape, floptical, etc.
- Multiple I/O bus support (SCSI, NUbus, PCMCIA, etc.).
- Optical jukebox IFS.
- IFS, subsystem kits.

- Device controller sharing (see floppy, parallel, serial, etc.).
- Make driver code pagable.

Video:

- Support for multiple active desktops.
- Support for multiple active video adapters.
- 3D graphics (SGI OpenGL).
- Additional standard transforms.
- Scalable hi-color/hi-res support (16 color to 64K color, 640x480 to 1600x1200).
- Support for Device Independent Color.

Setup.

- Consistent install, setup, configure & update model for devices (in Control Panel).
- Easier setup/install for 3rd party drivers.
- Consistent generic interface to load/unload drivers in the Control Panel.

Serial/Parallel

- Redesign drivers to support plug/play protocols (standard serial, x.25, ISDN, etc.), sort of like how the net layers bind together This allows vendors to just be able to plug in their protocol rather than have to replace our entire driver
- Class/port/miniport design to accommodate multiple devices and pass through (backup, XIRCOM, mini-SCSI, MultiMedia audio, print, etc.).
- Complete consistent multiple port support, including Control Panel.

Input Devices:

- Multiple input devices (keypads, keyboards, tablets, pens, membrane pads, digitizers, etc.) active.
- Built-in support for additional input info (temp, pressure, XYZ 3D motion, angular motion, proximity, extensible other)

Printing

- Extend the Forms database to have a metafile associated with them. This provides better performance. It allows the system to send the complete metafile to the printer one time only and to send a small command per page. Applications can display the metafile on the screen in a true WYSIWYG manner and print it very efficiently on all major printers.
- Additional improvements beyond covering more of the market are TBD.

MultiMedia

- Improve what is in the Windows NT 3.1 DDK. Details are TBD.

Benefits

- *Ease of use*. If we provide better setup capability and better market coverage, we provide better ease of use for our customers, both end-user and developer.
- *Performance*. Driver speed will be improved.

Compatibility

- All Windows NT 3.1 drivers are compatible.

Non goal

- New features in the Cairo drivers will not run on Windows NT 3.1.

Under consideration

Some of the work mentioned above may be in the NT v1.1 product. As its definition is more complete, this section will be updated.

Several vendors have asked that NT be able to

- boot without INT 13h. NT requires the IHV to write a separate INT 13h driver just for boot up. The driver is not used again. IHVs (especially SCSI developers) do not want to have to go to the time and expense of writing this.
- boot without INT 10h or EGA/VGA. NT requires that you have EGA/VGA to boot. This can cause someone to have whatever their video card is plus a VGA card for boot. Vendors have asked that we change setup to not require VGA.
- boot from any storage device like CDROM, tape, floptical, etc. Compaq has requested that we allow booting from CDROM. This would eliminate the need to ship the two boot floppies in each box.

As part of supporting multiple input devices, the following need to be considered:

- multiple pointing devices active on desktop
- muluple cursors active on desktop

To help make setup and installation easier for 3rd party drivers, we may want to consider using resources in the driver binary to describe driver dependencies/capabilities and/or do away with .INF files altogether.

VII. Requirements: Supplemental Services

Multimedia Components and Services (Clockwork) - Jim Green

Clockwork will enable and facilitate the development of applications for capturing, authoring and playing multiple synchronized streams of time-based data by providing generalized synchronization and data manipulation services. Clockwork will also provide the capability to install media processing components which perform data manipulation functions such as compression or decompression. These components will be defined and implemented as "objects" using the Component Object Model (COM). Standard interface definitions for Clockwork's components are designed to work in coordination with one another within the Clockwork architecture.

Goals

Clockwork services will ensure a consistent user experience when rendering multimedia data, independent of the specific machine used to process the data.

Clockwork services will make it easier for developers to create applications which need to synchronize and manipulate multimedia data based on events and time.

Clockwork will enable ISV's to provide better tools and applications, by allowing them to concentrate on features and ease of use rather than on solving the technical problems of coordinating and synchronizing the data. These services must be robust enough to address the needs of the professional high-end media producer.

Clockwork will enable IHV's to provide hardware acceleration for multimedia processes and to expose their functionality in a consistant way.

Clockwork will provide an architecture for support of distributed multimedia communications including audio and video teleconferencing and multimedia electronic mail in heterogeneous computing environments.

Benchmark

Clockwork will provide better performance than Video for Windows under similar load conditions, but for a wider variety of time-based data formats. In addition Clockwork will provide for more graceful degradation of performance due to system constraints.

Features

- *Clock Services.* Clockwork deliverables will include a *logical clock* that allows applications to manipulate a collection of media processing components based on a common view of time. The clock component uses a standard interface for starting and stopping the clock, getting and setting time, and changing the rate at which time flows. These clock services are independent of the actual source of the timing information which is provided by actual physical clocks. These physical clocks are encapsulated by installable components which provide access to an actual physical device that is capable of providing timing information to the logical clock. Examples of these devices are Win32 timer services, and the clock(s) associated with a multimedia device such as an audio card, or a VISCA device. Allowing these clock components to be installable enables third parties to provide access to clocks on their hardware that may have a higher resolution than the standard Win32 timer services.
- *Installable Media Processing Components.* Media processing components (called *filters*) are the objects in the Clockwork architecture that actually manipulate the multimedia data. Examples of filters are: audio or video compressors or decompressors, audio or video rendering devices (i.e., sound card or graphics accelerator), media digitizers, etc. Allowing these components to be installable enables third parties to provide handlers for new media types and formats more easily, and also enables hardware vendors to accelerate multimedia processing with special hardware (e.g., a DSP).
- *Standard Control Interfaces.* The Clockwork system includes a set of standard interfaces for controlling multimedia processing components to facilitate their inclusion in applications. This feature also provides for the integration of media from different vendors in a single presentation.
- *Standard Degradation Interfaces.* Multimedia data differs from more traditional data in two ways: the volume of data that needs to be processed, and the real-time constraints on the processing of that data. By providing standard degradation interfaces, Clockwork allows applications to define specifically what should be done when system resources are constrained, without having to know all of the details about the media formats.
- *System Support Functions.* In addition to the core components, Clockwork will provide a set of support functions to simplify the development of applications and installable components. These support functions include: services for installing and registering multimedia processing components and their characteristics, services for specifying the flow of data through a set of multimedia processing components (called a *filter graph*), services for saving and restoring the configuration of a graph, and a collection of pre-coded modules to make it easier for I*V's to build their own filters.

Benefits

Benefit of Clockwork for Hardware Vendors

The Clockwork architecture takes into account the fact that there is much room in the current generation of PCs for hardware acceleration of multimedia functionality. Clockwork addresses this need by providing for installable filters and installable clock components which encapsulate the underlying implementations, thereby providing a system in which building blocks can be implemented either in hardware or software. Additionally, communication between these modules may be implemented in a way that isolates the acceleration hardware from the central processor with respect to the data flow.

Benefit of Clockwork for Software Vendors

By expanding the set of standard multimedia services to enable synchronization, temporal abstractions, and a more flexible data flow architecture, applications may control more complex scenarios without having to deal with low level implementation details. Thus allows application writers to concentrate more on what an application should do, rather than on details that do not inherently add value to their application. For example, Clockwork's standard degradation and control interfaces allows the same application to run on a very fast machine (or machines with special purpose hardware) providing all of the resolution of the media, but also allows the application to run on a slower, less capable machine, by reducing the resolution of the media according to a strategy choosen in advance by the developer.

Advantages of Clockwork for End Users

Clockwork is a system software architecture, and as such is not directly visible to an end user. However, by raising the levels of abstraction and flexibility, we allow application writers to concentrate more on adding functionality to their applications than on implementing system software. This should translate into better, easier to use applications that do more for the end user.

Compatibility

Clockwork will provide a set of standard media processing components that capture/render the media types currently supported by the Win3.1 multimedia services, Video for Windows and the multimedia services for Chicago. However, the storage model for Clockwork will be based on ISTORAGE rather than directly on the AVI/RIFF formats. (This migration may take place in the implementation of the multimedia services for Chicago, but if it is not done there, it must be done as part of the Clockwork deliverables.)

The Clockwork architecture will be designed to optimize the synergy with other Microsoft system software components such as OLE 2, ISTORAGE, and Visual Basic and with strategies for cross-platform support such as WOSA

Non goal

Clockwork will not provide multimedia "tools" such as authoring tools, script interpreters, capture/playback applications, etc., except (perhaps) as sample code. It is assumed that these types of tools and applications would be clients of Clockwork services.

Under consideration

This section describes only the Clockwork multimedia system and its components. The Multimedia Technology Group is still evaluating the services it may need to provide to Cairo for supporting applications developed for the existing "Media Control Interface (MCI)" architecture.

Index

Edit history

Date	Who	Version	Reason
10/20/92	Daniel Conde		Create
11/10/92	Daniel Conde	1	First release to group. Limited distribution within Cairo
11/30/92	Daniel Conde	2	Interim revision for limited distribution. Added a longer overview.
12/3/92	Daniel Conde	3	Interim revision for limited distribution. Reduced size of overview. Added continuity section.
12/15/92	Daniel Conde	4	Solid draft. General agreement within Cairo; open issues remain. Incorporate overview changes from Paul Goode. Fix enhancements section for Windows NT P1 and P1.1 issues. Split packaged product and SDK documentation. Reorganize chapter order. Add DDK section. Remove Messaging temporarily. Distribution to Jim Allchin and his direct reports.
12/17/92	Daniel Conde	5	Copy edits to Version 4. Add Comm and RPC section. Put ClockWork section in.
12/18/92			Bill Gates review distribution canceled.
2/17/93	Greg Lobdell	6	Rewritten Overview, new organization for entire doc. New sections on Delphi authoring, InfoDoc, Mail Server, Comm/RPC, and Multimedia. Review to all Cairo Leads
3/5/93	Greg Lobdell	7	Clean-up of issue raised from V6– virtually all sections changed. Added Infodoc, Basic programmability sections. Widely distributed to MS Systems Division.

Appendix 1 - Terminology

This section describes terminology and concepts in Cairo. We will use terms that are consistent with Windows 3.1, Windows 3.1 NT, and Microsoft Dictionary.

C++ - a popular object-oriented language based on the systems programming language C.

Cairo - Code name for a new Microsoft Windows operating system.

Class Library - a set of object-oriented classes for reuse by programmers. Typically these classes provide default standard implementations and also provide a structuring of code for easier understanding.

Code Library - a set of standard function implementations for reuse by programmers. Typically these function implementations implement system-standard policies to make interoperability and correct functionality easier for a programmer to achieve.

Component Object Model - a standard way of using code and interfaces in an object-based style defined by the Object Linking and Embedding (OLE) product. The model defines how objects are created, named, defined, invoked, destroyed, and connected.

Components - Cairo or OLE objects that have been specifically designed to plug into aggregate object compositions. Typically, a component performs one specialized activity and exports one or more interfaces that categorize and describe the semantics of this activity. A button object, for instance, performs the act of signaling other components when pressed, and exports a *buttonEvents* interface which might consist of `buttonPressed` and `buttonReleased` methods. Component objects guarantee support for infrastructure that includes persistence, self-description, and runtime support for dynamic connect/disconnect to other components.

Connector - Connectors, like properties, are intrinsic parts of the object to which they belong. They can be referred to and located by name. Since there are two sides to a connection, there are two kinds of connectors: plugs, and jacks. From the perspective of the component, plugs represent interfaces implemented by the component itself, and jacks represent interfaces for which the component can be a client.

ConstructionSite - A Cairo container that is an editor for composite components. Object compositions are created within a constructionSite; these compositions, when finished, are packaged themselves as components.

Context Menu - a list of operations available on an object. It is materialized in response to a right mouse button click on any end user object.

Data Flow - a graph-based algorithm expression paradigm in which graph *nodes* represent functions and graph *arcs* represent data paths.

Digital signature - a token that accompanies a digitally signed message and allows the receiver to *verify* the identity of the sender. Typically digital signatures are generated by first computing a *message digest* and then using a public-key algorithm to encrypt the digest with the sender's private key.

Downlevel systems - Systems, data formats, and conventions used by pre-Cairo systems. For example, the DOS FAT filesystem is a down level file system.

Object composition - A collection of objects and bindings between their interfaces.

OLE 2 - Object Linking and Embedding 2. This is the second version of a product designed to assist application developers in creating interoperable applications. It uses the component object model and defines a fixed set of interfaces oriented toward the creation and maintenance of compound documents.

Pluggability - A component's willingness to support well-factored interfaces, persistent storage of bindings, and dynamic connection to other components.

Privilege Attribute Certificate (PAC) - a data structure that is used to encapsulate a principal's identity and the authorization information like group memberships and privileges etc. These PACs are sealed into service tickets issued by Kerberos.

Proxy - a token that allows one to operate with the rights and privileges of the principal that granted the proxy. Naturally it must be possible to verify that a proxy was granted by the principal that it names and that it has not been tampered with.

User Object Model - the gospel of objects according to the end user.

