Why DCOM on non-Microsoft Platforms?

EntireX DCOM brings the DCOM programming model to non-Microsoft platforms in a manner that is true to the original programming model and wire protocol. DCOM offers sufficient technical and semantic advantages over existing communications solutions to justify the effort of bringing it to platforms such as UNIX (IBM S/390 UNIX System Services, formerly Open Edition, also supported).

EntireX DCOM allows transparent access to in-process, local out-of-process or remote objects in a heterogeneous distributed environment. Like the Windows NT and 2000 release, EntireX DCOM has support for object naming with monikers, hierarchical object storage using compound files, binary reuse via COM aggregation, and accessibility from interpretive/scripting environments via Automation.

EntireX DCOM provides the object-oriented systems-programming services available in DCOM in a manner that is true to the original programming model, and yet interacts properly with native programming styles and idioms.

What does EntireX DCOM consist of ?

EntireX DCOM is an implementation of the non-graphical part of Microsoft's DCOM technology. EntireX DCOM allows DCOM-enabled applications running on non-Windows platforms to communicate with DCOM components in a heterogeneous network, i.e. a network with mixed architectures. So, for example, EntireX DCOM allows DCOM-enabled server applications running on UNIX or UNIX System Services to be accessed by DCOM clients on Windows NT and 2000, and vice versa.

EntireX DCOM consists of the following main parts:

- Runtime environment
- Software Development Kit (SDK)

Runtime Environment

In order for DCOM applications and components to run and communicate, it is necessary to have an EntireX DCOM runtime environment. This consists of a set of shared libraries (DLLs) and tools to monitor and manage the DCOM multiuser environment.

Software Development Kit (SDK)

The SDK is a collection of tools, libraries and examples which enable the DCOM developer to produce runnable DCOM applications.

Functionality Supported by EntireX DCOM

The following aspects of DCOM and Win32 are supported:

Win32 System Service APIs

- Threads, synchronization objects (mutexes, semaphores, events and critical sections)
- File I/O
- Dynamic library loading etc.
- System registry API, with a port of the Windows regedit utility
- Security APIs (Security Descriptors, ACE, ACL, Token, Impersonation).

EntireX DCOM supports a subset of the Win32 APIs that can be used by applications.

Microsoft Interface Definition Language compiler (MIDL)

- The midl compiler, version 3.00.44.
- Standard marshaling with midl-generated proxies and stubs.

DCOM Basic Functionality

- The COM task allocator.
- Both the Single-Threaded (STA) and Multi-Threaded Apartment (MTA) models.
- Adding entries to the class table.
- Instantiating objects both in-process and out-of-process, locally and remotely.
- Connection Points.
- RPC with static/dynamic endpoints (dynamic via endpoint mapper).
- Loadable RPC transports to allow RPC/DCOM to utilize different protocols.
- Local procedure call mechanism for fast local inter-process communication.

Structured Storage and Compound Files

- Creating/opening compound files (StgCreateDocfile, StgOpenStorage).
- Creating/manipulating storage and stream elements (IStorage and IStream interfaces) of compound files across process boundaries.
- A console-based document file viewer utility "stgview".

Naming and Binding (Monikers)

- Registering objects in the running object table (ROT).
- Creation and composition of basic moniker types (file, item, composite, anti).
- Stringification/destringification of basic moniker types (MkParseDisplayName).
- Binding file and composite monikers. This works locally and also from Windows NT and 2000 to UNIX, provided that you are using a tool (such as Samba) that allows you to mount a UNIX file system on Windows NT and 2000. The feature is not currently available from UNIX to Windows NT and 2000.
- GetInstanceFromFile. This works locally and also from Windows NT and 2000 to UNIX, provided that you are using a tool (such as Samba) that allows you to mount a UNIX file system on Windows NT and 2000. The feature is not currently available from UNIX to Windows NT and 2000.
- FileMoniker activation using ActivateAtStorage. This works from Windows NT and 2000 to UNIX, provided that you are using a tool (such as Samba) that allows you to mount a UNIX file system on Windows NT and 2000. The feature is not currently available locally or from UNIX to Windows NT and 2000.
- A utility "irotview" to view the running object table.

Automation

- IDispatch interface implementation and supporting API functions.
- The mktyplib type library generator.

Active Template Library

- This kit provides an implementation of the Microsoft Active Template Library (ATL) version 3.0 that works with EntireX DCOM.

Miscellaneous

- The uuidgen and guidgen utilities for GUID generation.

Current Limitations, Differences, Unsupported Functionality and Hints

Some aspects of COM have known limitations or are currently unsupported. These are listed below. Where appropriate, hints on achieving an alternative solution are provided.

General

- Asynchronous storage, PropertySetStreams and PropertySetStorages. These Windows NT 4.0 and 2000 features of compound files are not yet supported.
- The only well-tested RPC transport available with EntireX DCOM is TCP/IP.
- There is currently no support for remote activation of in-process server DLLs (i.e., surrogate servers).
- Non-midl generated interface marshalers are not supported.
- Interoperability with Windows 95/DCOM is not supported.

Share Names and File Names

All of the EntireX DCOM programs and libraries work with path names up to 255 characters long.

On Windows NT and 2000, file names are always translated to upper case, so various algorithms must be used on the non-Windows side to find certain files.

Share names are resolved as follows on non-Windows platforms. There is a key HKEY_LOCAL_MACHINE\Software\Software AG\DCOM\ShareNames in the registry. Below this you can supply a share name as a new subkey, for example HKEY_LOCAL_MACHINE\Software\Software AG\DCOM\ShareNames\foo = "/fs0701/prog/bin/foo".

If a name in Universal Naming Convention (UNC) format is sent as a file name from Windows NT and 2000 to a non-Windows platform, for example \\ABC\FOO\V1\FOO.BAR, the following steps are taken to resolve the name:

1. ABC is the machine name, and a check (case insensitive) is made that this is the correct machine.
2. FOO is checked to see if it is a key under ShareNames in the registry. If so, then FOO is replaced by the path in the registry.

3. If there is no share name in the registry, a check can be made to see if the name is the home directory of a user. This check is made if the value of AllowUserHome, which is also under HKEY_LOCAL_MACHINE\Software\Software AG\DCOM\ShareNames, is "Y". The default value for AllowUserHome is "N".

4. If a path is found, it is combined with V1\FOO.BAR. All backslashes ("\") are replaced by slashes ("/"), and the following algorithm is used to try to find the file ("/fs0701/prog/bin/foo/V1/foo.bar"):

5. If the path exists exactly as originally specified (case sensitive), it is used.

   If a path is found ("/fs0701/prog/bin/foo/V1/foo.bar") using the algorithm just described, an attempt (case insensitive) is made to complete the path, for example "/fs0701/prog/bin/foo/v1/FOO.BAR". The path must be unique (case insensitive). An error would result if, for example, another directory "/fs0701/prog/bin/foo/V1" exists.

Security

A Security Service Provider (SSP) package (libsecurity) is provided. It is a fully-functional LAN-manager-based SSP that supports secure communication between Windows NT and 2000 and non-Windows platforms at the DCOM level.

Unicode (wide character strings)

Internally, DCOM uses a 2-byte UNICODE character representation. On platforms where UNICODE support is not provided, EntireX DCOM maps UNICODE to wchar_t. Incoming 2-byte UNICODE characters are sign-extended to the wchar_t size and outgoing wchar_t are truncated to 2-byte UNICODE as appropriate.

The OLECHAR type is defined to match the wchar_t support from the native compiler. If you store OLECHARs in files or opaque byte arrays, be aware of this difference. The OLESTR macro works as it should, however. The following table shows the wchar_t sizes for various platforms:

Platform	wchar_t size
AIX	2 bytes
Tru64 UNIX (formerly Digital UNIX)	4 bytes
HP-UX	4 bytes
Sun Solaris	4 bytes
IBM S/390 UNIX System Services	2 bytes
Linux	4 bytes

RPC Data Conversions

The DCOM wire protocol makes use of the principle "the reader makes it right". Thus, typically the initiator of an RPC sends the data on the wire in its natural representation

(little/big endian, ASCII/EBCDIC, IEEE/IBM float). The RPC header contains information on the format of the data it contains. The receiver can then examine the RPC header to determine whether the data received is in the same format as its own and, if required, convert it into its own format. In EntireX DCOM the following conversions take place:

- On big-endian machines (e.g. Solaris/SPARC, HP-UX/PA-RISC, AIX/RS6000, S/390) little-endian data received is converted transparently into big-endian.
- On little-endian machines (NT/Intel, DEC-UNIX/Alpha) big-endian data received is transparently converted into little-endian.
- On machines using ASCII character encoding, ASCII single-byte characters received are not converted. Note, however, that due to a restriction in MS RPC (based on DCE RPC) which does not allow for codepage information to be specified in the RPC header, it is implicitly assumed that both sides of a DCOM communication are using the same codepage. EBCDIC characters received are converted to ASCII ISO-8859-1 under the assumption that the characters are encoded in EBCDIC codepage 1047.
- On machines using EBCDIC character encoding, ASCII single-byte characters received are converted to EBCDIC codepage 1047 under the assumption that the characters received are encoded in ASCII ISO-8859-1.
- For wide characters it is implicitly assumed that they are encoded in UNICODE. Microsoft DCOM under Windows NT and 2000 has no provisions to deal with any wide-character encodings other than 2-byte UNICODE. Thus EntireX DCOM (on all UNIX and S/390 platforms) converts wide characters using the following rules:
    - 4-byte-wide characters are narrowed to 2-byte. The upper half of 4-byte-wide characters is lost.
    - Codes in the range 0-255 are converted using the rules for single-byte characters.
    - Codes above 255 are not converted at all (neither ASCII nor EBCDIC)
- Under S/390, IBM float and double data are converted to and sent on the wire as IEEE float and double values. When IEEE float and double values are received under S/390 they are converted into IBM float and double format.

MIDL

- The EntireX DCOM MIDL compiler corresponds to Microsoft midl version 3.00.44.

- The runtime currently does not support highly optimized midl-generated code. Options beginning with "-Oi" are not supported.

- The midl compiler (and the NDR engine) currently do not support stubless proxies (option -Oicf). Use the -Os option to generate compiled proxy/stub pairs. If you are porting code that assumed -Oicf-compiled proxy/stub pairs, be aware that -Os marshalers do not zero-fill output parameters. Support for -Oicf is planned for a subsequent release.

- If you implement custom marshaling, be aware that sizeof(wchar_t) is platform-dependent (see the above wchar_t sizes table). Furthermore, some machines are big-endian (high order byte first). Use the pickling feature of midl if you want to serialize portably. If you use standard marshaling via midl-generated proxy/stubs, these differences are addressed in the NDR runtime.

- The default preprocessor used by EntireX DCOM's midl compiler is the platform's native C++ compiler. See the detailed table in EntireX Platform Coverage.
  Microsoft uses its C/C++ compiler (cl.exe) as the default preprocessor, hence the __cplusplus constant is defined during the preprocessing pass of midl compilation. If this causes problems, we suggest that you use the preprocessor constant MIDL_PASS to exclude critical code sections, as in:

  ```
  #ifndef MIDL_PASS
  ...
  #endif
  ```

  The default preprocessor arguments used are:

Platform	Arguments
AIX	-E -+ -DMIDL_PASS
Tru64 UNIX	-E -x cxx -DMIDL_PASS
HP-UX	-E -DMIDL_PASS
Sun Solaris	-E -DMIDL_PASS
IBM S/390 UNIX System Services	-E-+ -DMIDL_PASS -Wc,NOLOC
Linux	-P -E -xc -DMIDL_PASS

  The default preprocessor can be overridden from the command line. For example, the GNU C/C++ compiler (gcc version 2.95.1 or later) can be used on Linux platforms with the following command line (assuming that gcc is installed in /usr/local/bin):

midl -cpp_cmd /usr/local/bin/gcc -cpp_opt '-P -E -xc -DMIDL_PASS'

- Files generated by EntireX DCOM's midl have to be compiled with the additional compiler switches -DSAG_COM and -D<platform specification> to activate the platform-specific code sections (<platform specification> is defined as appropriate in $EXXDIR/$EXXVERS/include/makefile.incl, variable C_DEFINES). In addition you have to specify the directory path of the EntireX DCOM header files with the -I switch.

- Midl supports a new option "-env UNIX", which is used to specify that the target environment is UNIX. This option is set by default.

- By default, wide characters are supported; the data representation for a wide character is platform dependent.

- Reading midl options from a file using "@<filename>" is not implemented, since it is not necessary on UNIX systems, where there are few limitations on the length of the command line.

- Absolute file names, i.e. file names beginning with a slash ("/"), must be preceded by a backslash ("\"), e.g. "\/FS/fs0", otherwise the names are interpreted as "/<x>" options.

- Proxies need additional compiler flags. See the section "Generating Proxy/Stub Libraries (DLLs)" in the SDK description later in this document.

- For HP-UX pragma midl_echo is not supported by midl. It is recommended to use cpp_quote instead in the idl files.

guidgen

- guidgen is a command line utility - there is currently no GUI version.

- guidgen functionality is restricted to the definition of GUIDs (the portion that is marked with DEFINE_GUID in the Windows version). This functionality is activated with the -d command line argument. Without this argument, guidgen behaves identically to uuidgen.

uuidgen

- The version number is not 1.00 but 1.01 to reflect modifications to the Microsoft tool.

mktyplib

- The command switch '/' (slash) is not supported. Use '-' (hyphen) instead.

- The default preprocessor is the platform's native C++ compiler, as for midl.
- The default preprocessor options used are:

Platform	Options
AIX	-E -+ -D__MKTYPLIB__
Tru64 UNIX	-E -x cxx -D__MKTYPLIB__
HP-UX	-E -D__MKTYPLIB__
Sun Solaris	-E -D__MKTYPLIB__
IBM S/390 UNIX System Services	-E -xcxx -D__MKTYPLIB
Linux	-P -E -x c++ -D__MKTYPLIB__

- Note that the default preprocessor may be overridden from the command line. For example, the GNU C++ compiler (gcc version 2.95.1 or later) can be used. To use the GNU compiler (assuming that it is installed at /user/local/bin), use a command like:
  mktyplib -cpp_cmd /usr/local/bin/gcc -cpp_opt '-P -E -x c++ -D__MKTYPLIB__
  '

makedef

- The makedef utility does not currently support renaming exported functions, which is possible under Windows NT and 2000 with the syntax "DLL_xx=xx".

Active Template Library (ATL)

- The original Microsoft ATL uses default template arguments, which are not currently supported by the compilers available on any of EntireX DCOM's platforms. The SDK contains a modified version of the ATL, version 3.0.
- This library cannot currently be used on Linux because it relies on static data members in class templates, which the GNU compiler does not yet support. The ATL will be ported to Linux when support for this C++ feature becomes available in the GNU compiler.

Online Help

Usage information for each utility is available by running the utility with the "-h" command line option. There is no additional online help provided with this release.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.