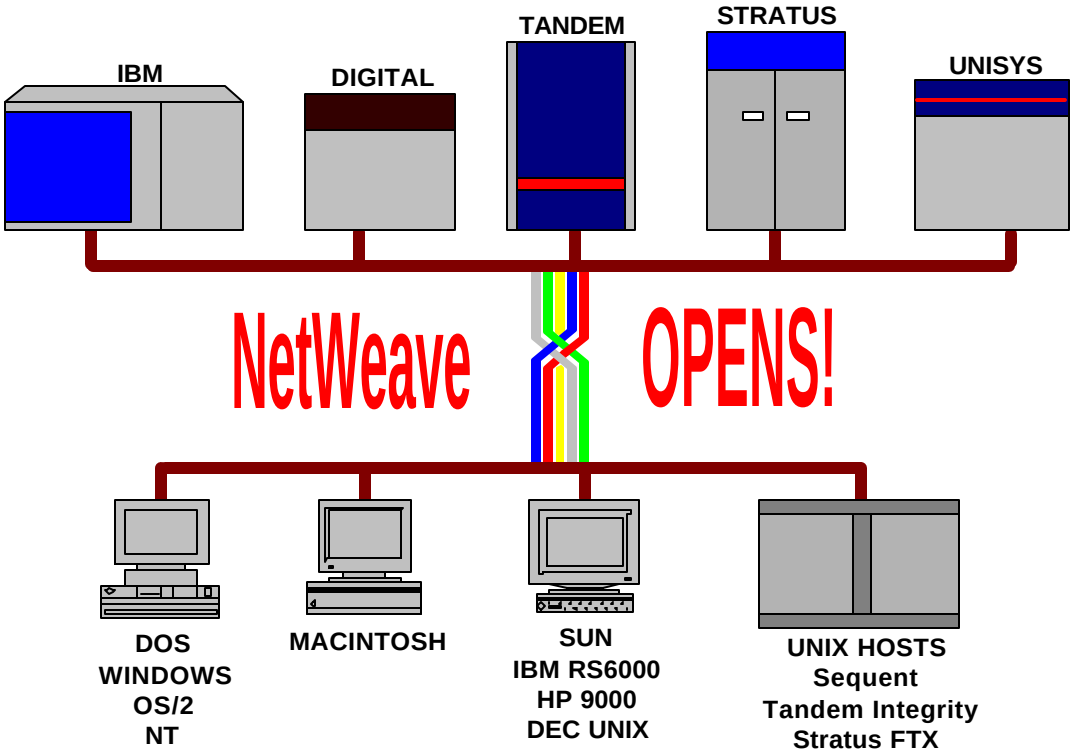


NetWeave

The Application Network

NetWeave - The Application Network

Today's enterprise-wide, mission critical applications require legacy and open systems to cooperate in complex, distributed applications. A web-based order entry system, for instance, may need to obtain customer information and to update billing information held by an IBM financial system. It may also need to access and update inventory on a Stratus warehousing system and to request new inventory from a Tandem scheduling system. The Tandem, in turn, may have to request the manufacture of items from a OpenVMS-based manufacturing control system.

NetWeave provides the interoperability services required to build such distributed applications. Using standard protocols, NetWeave links your system to a heterogeneous network of applications and data bases. Your applications can reach to the remote data and processes of IBM, Compaq Himalaya and OpenVMS, Stratus, Unisys, and UNIX hosts and Windows 9x/NT/2000 and UNIX workstations. NetWeave is the first - and after more than a decade of successful use, still the only - product that provides you with such a full range of platforms and services.

NetWeave's powerful library of remote services provides software developers with data query and update capabilities wherever that data resides and in whatever form, from proprietary file systems to SQL data stores. Additional features provide peer-to-peer messaging services to link applications across the network, thereby allowing developers to build their own client/server and peer pairs.

NetWeave enables truly interactive applications to expand beyond your system into heterogeneous environments. Use NetWeave to break your connectivity barrier. It's the perfect link between your system and the other systems you may use.

NetWeave's Middleware Services

NetWeave's middleware services support both the client/server and the peer-to-peer distributed application models. The client/server architecture is the model of choice for interactive applications, whereas the peer model supports work flow applications.

The advantages of these distributed models are many:

- Distribution - Clients, servers, and peers may reside on different platforms.

- Scalability - Additional platforms and functions easily may be added to an expandable application network base.

- Reliability - Failed servers or peers may be replaced automatically with surviving systems.

- Performance - High volume, highly interactive applications can be implemented.

To support these models, NetWeave provides a broad range of interoperability services:

- **Client/Transaction Services** allow a client application to send a complex, interactive transaction to a remote application server and receive an immediate response.

- **Message Queuing Services** guarantee delivery of messages even in the face of server or network failures.

- **Broadcast Services** provide reliable distribution of information to remote applications.

- **Client/Database Services** allow a client application to directly access and update data in foreign proprietary file systems and in SQL databases.

- **Reliable File Transfer Services** provide efficient bulk transfer of data around the application network, with the ability to recover from source, destination or network failures without restarting the transfer

- **Batch Services** allow applications to execute commands on remote systems.

- **Transaction Services** ensure consistency of updates for remote data store updates.

- **Data Conversion Services** reconcile data format differences between platforms.

- **Security Services** ensure controlled access to the application and its data.

- **Directory Services** allow data and servers to be freely moved about the network.

- **Routing Services** route NetWeave messages between systems even if those systems use different communication protocols.

- **Dual Channel Services** support redundant communication paths between application network components.

- **Remote Spooling Services** allow a client to take advantage of a remote spooler to deliver text files to a remote printer, file, or application.

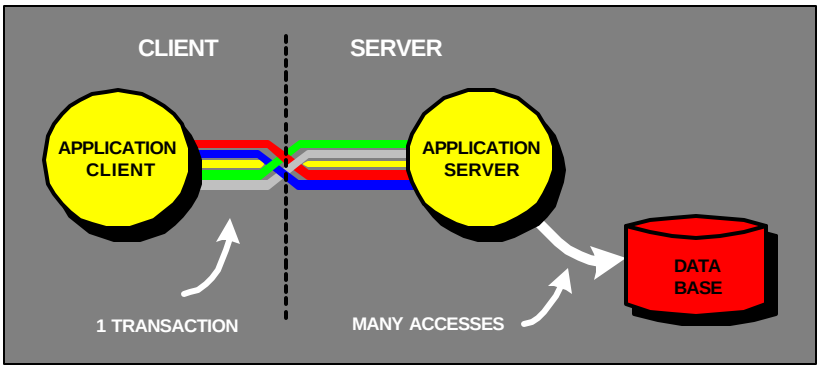
More detail about these services can be found in the following paragraphs.

Client/Transaction Services

Client/transaction services are favored for high performance OLTP (on-line transaction processing) applications. Typically, the application program is split between a workstation client and a host server, though this model also applies to host-client/host-server and host-client/workstation-server architectures as well.

The client side of the application is generally responsible for presentation services, editing of incoming data from the user, and formatting of outgoing data to the user. The server side is responsible for the business logic governing database access and update.

The client/transaction model is highly effi-cient since a single transaction sent over the network can en-compass a significant amount of database activity. However, the application must be totally defined in advance. This is, of course, the realm of OLTP.



NETWEAVE CLIENT/TRANSACTION SERVICES

These services are also available for peer-to-peer application communication in which two applications must communicate as equals. Either application may send a message to the other with or without the requirement for a reply.

Message Queuing Services

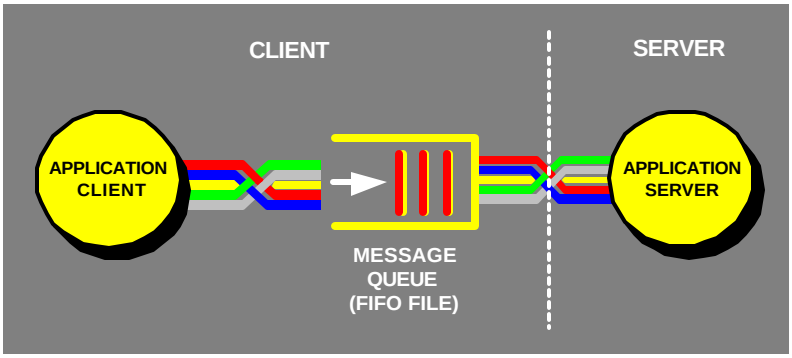
The previously mentioned services are highly interactive. When the client makes a request, it expects an immediate response from the server.

However, many distributed processing requirements involve communication between peer applications, in which neither application is a server to the other. These applications do not require interaction; instead, they require a guarantee that each message will be delivered ultimately to the receiving application even if the receiving platform or communication network is currently down.

Typical examples of these applications are:

- an order entry system that must pass invoicing and receivable transactions to a financial system.

- a train control system that passes trains to a neighboring rail system.

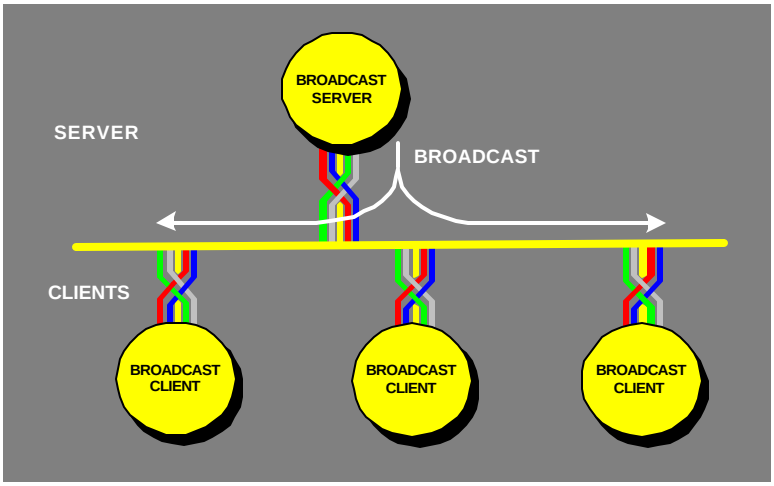


NETWEAVE MESSAGE QUEUING SERVICES

This is the realm of message queuing services. With Net-Weave's queuing ser-vices, each message is safely stored in a FIFO (first in, first out) file by the sending applica-tion, from which it can be read at the receiver's convenience. The FIFO file may be located on any platform in the network. It is typically located on the sending platform or on some other fault-tolerant platform in the network.

Broadcast Services

In many applications, information must be distributed to a large community of users. Often, this information is critical (such as stock quotations): and message delivery must be guaranteed.



NETWEAVE BROADCAST SERVICES

Broadcasting facilities are fundamentally one way and are therefore unreliable. A server broadcasting a message to a variety of clients has no way of knowing whether each client has missed any messages. Message delivery is usually guaranteed only if the server has direct connections to each client and is sending independent, though identical message streams.

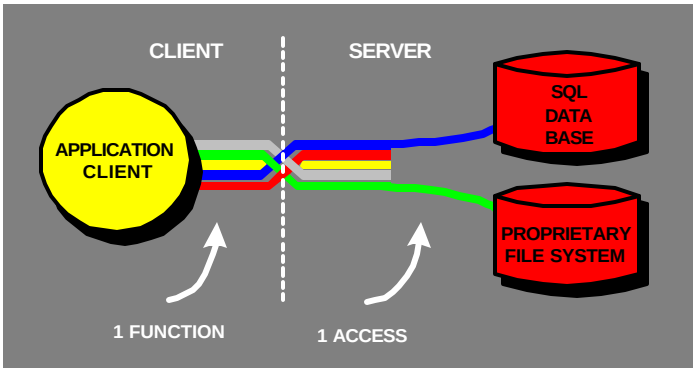
NetWeave's broadcast services maintain the efficiency of broadcasting a message only once but guarantee that each client receives all messages. It does this by giving each client a facility to detect missing messages. If a client determines that it has missed a message, it can use a reverse recovery channel to request retransmission of that message for its benefit.

NetWeave clients can register an interest in certain message classes and will only receive broadcast messages belonging to those classes.

Client/Database Services

Client/database services represent a highly flexible approach to ad hoc applications which cannot be clearly defined in advance. They are also useful when the database platform is closed to new application programs or when the cost or schedule for deploying new applications on the database platform is prohibitive.

Under this model, the application as a whole acts as a client to a remote NetWeave database server. The client application can manipulate records in proprietary files or can manipulate rows in SQL data bases managed by the database server just as if that data were local to the client platform.



NETWEAVE CLIENT/DATABASE SERVICES

The advantage of this model is that new applications can be written solely on the client system, with no knowledge needed of the inner workings of the database server. However, each database operation (open, position, read, update, select, fetch, etc.) must be sent individually over the communication network.

Client/database services are useful in many non-OLTP and light OLTP applications, such as:

- decision support requiring ad hoc data queries.

- prototyping OLTP applications before committing them to the client/transaction model.

- migrating applications, where applications must access data on other platforms during the migration process.

Reliable File Transfer Services

Often, the bulk transfer of files or tables is required across the network between heterogeneous systems. By their very nature, these transfers can be very lengthy. Nothing is more aggravating (and perhaps even catastrophic) than having the transfer fail in the middle and having to restart it.

NetWeave's Reliable File Transfer (RFT) solves this problem. It not only provides file transfer between different data bases, converting formats as it goes, but it also checkpoints its progress. Thus, if a transfer fails in the middle due to a network problem or processor failure, RFT will pick up where it left off when the connection is restored.

In addition, RFT's use of sophisticated data compression techniques ensures the maximum efficiency of data transfer, leading to shortened file transfer times.

In addition to the transfer of files, RFT also can transfer contents of FIFO queues from one platform to another.

Ancillary Services

Several important sets of ancillary services are provided by NetWeave to support the previously mentioned distributed application models.

- **Transaction Services** are used by a client application to bound a series of updates to a remote data base. By starting a transaction and subsequently committing it, all intervening updates are guaranteed to be made to the data base; or else none are made. Thus, the data base will always be left in a consistent state. Transaction protection is available for Client/Transaction, Client/Database and Message Queuing Services. It depends on using the transaction services provided by the target platform.

- **Data Conversion Services** are used to reconcile differences in data formats between diverse platforms. A NetWeave application can ask the type of a remote platform and then can use NetWeave's data conversion services to translate data from server format to client format and vice versa.

- **Security Services** offered by NetWeave provide controlled access to a server platform by using a challenge/response protocol. An application may provide its own level of security by incorporating a user-supplied security function. When a client application first attempts to use a server, it must provide a valid identification. It then must respond properly to a challenge issued by the security server. The challenge can be anything from a simple request for a password to a random token requiring a computed response.

- **Directory Services** allow a client to access a remote file, a remote table, or a remote process without having to know where that object is physically located in the network. The client application need only know the object by a pseudonym such as the PAYROLL file. When the object is opened by the client application, NetWeave will use the directory services to locate the object in the network and to determine the proper naming semantics for referring to that object.

- **Dual Channel Services** allow two systems to be connected by two communication channels. Should one fail, communication continues uninterrupted over the surviving channel. Upon restoration, the failed channel is automatically put back in service.

- **Routing Services** allow service invocations to be passed between two platforms which are not directly connected to each other, even if the platforms use different communication protocols.

- **Remote Spooling Services** capture data that is being sent to a spooler, despools that data, and forwards it to any remote foreign file for use by applications on the remote platform.

- **Batch Services** allow an application to execute a command or a program on a remote platform.

A Word about Interfaces

NetWeave supports a variety of programming interfaces suitable for integration into both the existing environments found on many Corporate IT host systems, as well as the new Web-oriented environments that many new IT development projects are insisting on:

- **C Language interface** : NetWeave is written in C/C++, so C on any platform can readily integrate with the NetWeave library.
- **COBOL Language Interface**. NetWeave provides a COBOL language interface for most supported platforms.

- **Platform-Specific Languages.** Many single-platform languages are supported, such as TAL on Compaq Himalaya platforms, ALGOL on Unisys A-Series, and PL/1 on IBM and Stratus platforms.
- **Java Interface**. NetWeave supports a Java interface implemented in a Java class library (Jar), Servlet, and Enterprise Java Beans environments.
- **COM+ Interface**. NetWeave has recently implemented a second generation COM interface which integrates smoothly in both MTS and non-MTS environments.

NetWeave Architecture

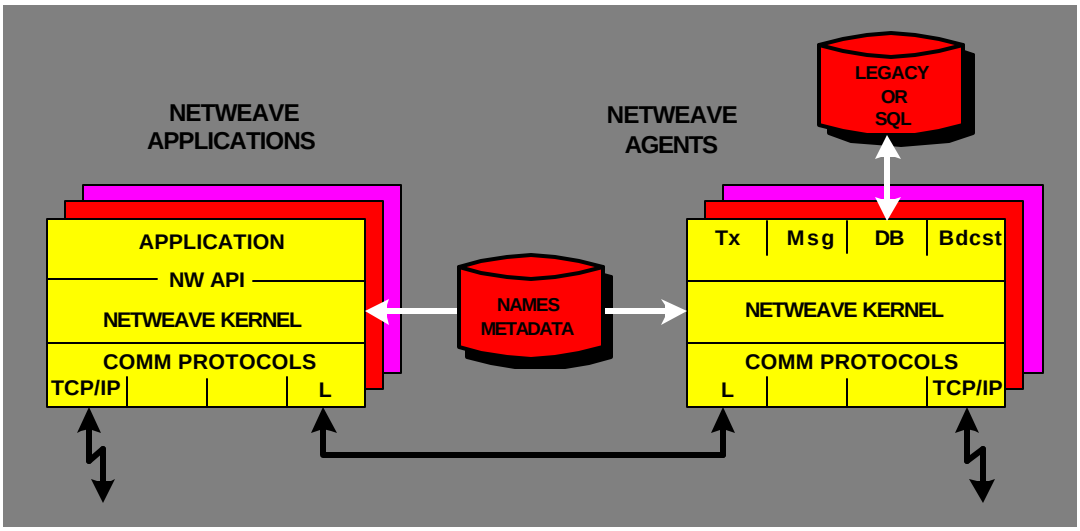
The NetWeave architecture is based on the NetWeave Agent. A NetWeave Agent generally resides on every computer system in a NetWeave application network.

The NetWeave Agent is a highly configurable multi-threaded server that provides the specific task or tasks required by an application. It comprises three main layers:

- The **NetWeave Kernel** provides all control, routing, data conversion, and service invocations required by the NetWeave Agent. It uses the NetWeave dictionary which contains object names and locations as well as the structure, or metadata, of network messages.

- The **communication layer** can have bound into it any number of protocol stacks (but at least one) for the various protocols that it must handle. NetWeave currently supports TCP/IP and the various local protocols of its supported platforms.

- Into the **services layer** is bound the particular NetWeave services required by the NetWeave Agent. These include transaction services (Tx), guaranteed messaging services (Msg), broadcast services (Bdcst) and remote database services (DB).



REMOTE NETWEAVE APPLICATIONS AND AGENTS

Any number of NetWeave Agents of any combination may be resident on a given system. In this way, NetWeave services may be partitioned across NetWeave Agent classes, each class containing multiple like agents for load sharing and balancing.

As with the multiple NetWeave Agents, there may be many NetWeave applications on a platform. A NetWeave application is a program with NetWeave's API bound into it. Interestingly, the procedures making up NetWeave's API are almost exactly the NetWeave Agent without the service layer. This includes the NetWeave Kernel and one or more communication protocol stacks. The NetWeave API may be used synchronously or asynchronously by an application. In addition, an application may be multi-threaded if its platform provides user or kernel threads.

One protocol that is common to both the NetWeave applications and the NetWeave Agents is the local protocol (L). This protocol is used for applications and servers to communicate with each other.

Some usage examples will serve to illustrate this architecture:

- A **transaction** is received by the NetWeave Kernel from its application and is routed. If its destination is to another application using a common protocol, then the transaction is sent directly to that application. This is the one case in which a NetWeave Agent is not needed. Otherwise, it is sent via the local protocol to a NetWeave Agent, which will route the transaction. Responses are returned via the same path.

- A **guaranteed message** put or fetch request or a database operation is routed to a local NetWeave Agent. If the message queue or data base is local, the

NetWeave Agent will execute the request directly. Otherwise, the Agent will route it to the appropriate remote NetWeave Agent and will return the data or status as applicable.

- A **network message** from a remote node destined for another remote node is received by the NetWeave Agent and is routed accordingly. Note that this is a case in which a NetWeave Agent configured with no services can act as a router.

NetWeave Customer Support

In addition to our hot line support for our NetWeave product, NetWeave specialists offer a broad range of services for planning and implementing distributed heterogeneous applications:

- Implementation of applications, either on a turnkey basis or as part of the customer team.
- Extension of NetWeave services to other platforms, operating systems, data bases, and communication protocols.
- Addition or enhancement of NetWeave's middleware services.

About NetWeave Integrated Solutions, Inc.

NetWeave is a product of NetWeave Integrated Solutions, Inc. based in Central New Jersey. NIS is a middleware integration specialist with specific focus on providing scale-able, high performance products and solutions within a variety of industries.

Reliability, versatility, and a wide performance range - these are the strengths of NetWeave and NIS.

NetWeave Integrated Solutions Inc.

GEDi Corporate Park
490 Rt 33W
Englishtown, NJ 07726
Ph: +1 732.786.8830
Fax: +1 732.786.8832
info@netweave.com

The NetWeave Application Interface

Messaging Services:

Client

EXECUTE	Create and run a remote server process.
IPC_CONNECT	Establish a connection with a remote server process.
IPC_WRITE	Send a request to a remote server process.
IPC_READ	Read a reply from a remote server process.
IPC_SHUTDOWN	Disconnect a remote server process.
STOP	Stop a remote server process.

Server

IPC_PUBLISH	Publish the server identification and await a connection.
IPC_ACCEPT	Accept a connection request from a remote client process.
IPC_READ	Read a request from a remote client process.
IPC_WRITE	Send a reply to a remote client process.
IPC_SHUTDOWN	Disconnect a remote client process.

Peer-to-Peer

IPC_CONNECT	Connect to a remote process.
IPC_READ	Read a message from a remote process.
IPC_WRITE	Send a message to a remote process.
IPC_SHUTDOWN	Disconnect from a remote process.

Broadcast

IPC_REGISTER	Register interest in a class of broadcast messages.
IPC_BROADCAST	Broadcast a message.

Message Queuing

FILE_CREATE	Create a FIFO file.
FILE_INFO	Return information about a FIFO file.
FILE_OPEN	Open a FIFO file for read only or write only access. Read only access may be in “peek” mode.
FILE_WRITE	Write a record to the tail of a FIFO file.
FILE_READ	Read the oldest record from the head of a FIFO file. Delete the record if not in “peek” mode.
FILE_POSITION	Delete or undelete records from the head (read) or tail (write) of a FIFO file.
FILE_CLOSE	Close a FIFO file.
FILE_REMOVE	Remove a FIFO file.

Information

IPC_OPTIONS	Return information from the communication layers.

Database Services:

Proprietary File Systems

FILE_CREATE	Create a remote file.
FILE_INFO	Get information about a remote file.
FILE_OPEN	Open a remote file.
FILE_POSITION	Position randomly in a file.
FILE_READ	Read a record from a remote file, with an optional lock.
FILE_WRITE	Write a record to a remote file.
FILE_UPDATE	Update the current record in the file, unlocking it if locked.
FILE_COPY	Copy a file from one system to another.
FILE_DELETE	Delete the current record in the file.
FILE_CLOSE	Close a remote file.
FILE_REMOVE	Purge and delete a remote file.
TRIGGER_REGISTER	Register a request to be notified of changes made to a file or table.
TRIGGER_READ	Return old and new data associated with a trigger event.
TRIGGER_CANCEL	Cancel a registered request for file change notification.

SQL Data Bases

SQL_CONNECT	Connect to a remote SQL data base.
SQL_SELECT	Select rows from a remote SQL data base.
SQL_FETCH	Return one selected row.
SQL_EXECUTE	Submit an SQL statement for execution.
SQL_COLUMN_BIND	Relate variables to columns of a row.
SQL_COLUMN_COUNT	Obtain count of selected items.
SQL_COLUMN_GET	Obtain name of a column.
SQL_COLUMN_INFO	Obtain name, data type of columns.
SQL_DISCONNECT	Disconnect from a remote SQL data base.

Other Services:

Transaction

TP_START	Start a transaction.
TP_COMMIT	Commit all the database updates since the last TP_START.
TP_ABORT	Undo all the database updates since the last TP_START.
TP_STATUS	Request the status of a transaction.

Data Conversion

SYSTEM_TYPE	Get the remote system type.
CONVERT_DATA	Convert a data field from one system's format to another.
CONVERT_RECORD	Convert a data structure from the source system format to the destination system format according to metadata stored in the directory's data dictionary.

Naming

INI_PUT_NAME	Put a new name in an application's user group.
INI_GET_NAME	Return the value of the specified name statement from an application's user group.
INI_DELETE_NAME	Delete a name statement from an application's user group.

Security

LOGON	Log on to a remote NetWeave service. Carried as parameters in the Connect or Open request.
PASSWORD	The password algorithm is specified and executed by the user-supplied security function.
LOGOFF	Log off of a remote NetWeave system. Carried as parameters in the Shutdown or Close requests.

Directory

GET_NAME	Convert a pseudonym for a file, process, or object to a node name and the correct name syntax for that node.

Miscellaneous

INIT	Initialize the NetWeave API in an application.
BATCH	Execute a command in a remote system.
SLEEP	Suspend the application until a defined event occurs.
SESSION_CLOSE	Release all resources associated with a NetWeave node.
EXIT	Shutdown NetWeave.
ERROR_TEXT	Convert an error code to an error message.
PING	Test a connection.

Structures:

CALL_BACK	The asynchronous call return. It is a structure pointed to by a parameter in most calls which contains context and a pointer to the completion function. If missing, the call is executed synchronously.
ITEM_LIST	Permits system-specific features to modify the action of a function call. It is a structure pointed to by a parameter in most calls which contains system-specific parameters. The structure is created via the following calls:
ITEM_LOAD_CHAR	Assign a character array to an item list.
ITEM_LOAD_LONG	Assign a long integer value to an item list.
ITEM_LOAD_SHORT	Assign a short integer value to an item list.
ITEM_LOAD_HANDLE	Assign a transaction identifier to an item list element.

Note: This is the API for NetWeave Distributed Services, which may not be compatible with older versions. Contact Netweave customer support at Vertex Industries before attempting to mix versions. All functions may not be available on all platforms without an engineering charge.