Appendix A: Constant Declarations and Structure Definitions

Information pertaining to Accounting Services

Constant Definitions

NWMAX_COMMENT_LENGTH	48
NWMAX_NUMBER_OF_HOLDS	32
NWMAX_OBJECT_NAME_LENGTH	48

Comment Types

NWAN_CONNECT_TIME	1
NWAN_DISK_STORAGE	2
NWAN_LOG_IN	3
NWAN_LOG_OUT	4
NWAN_ACCOUNT_LOCKED	5
NWAN_SERVER_TIME_MODIFIED	6

NWHoldInfo_t Structure

```
typedef struct {
        uint32    objectID;
        int32     holdAmount;
  } NWHoldInfo_t;
```

objectID contains the bindery objectID of the object submitting the hold.

holdAmount contains the amount that the objectID placed against the user's account balance.

Information pertaining to Bindery Services

Constant Definitions

NWMAX_MEMBER_NAME_LENGTH	48
NWMAX_OBJECT_NAME_LENGTH	48
NWMAX_PASSWORD_LENGTH	16
NWMAX_PROPERTY_NAME_LENGTH	16
NWMAX_PROPERTY_VALUE_LENGTH	128
NWMAX_SEGMENT_DATA_LENGTH	128

Bindery Object Types

NWOT_WILD	0xFFFF
NWOT_UNKNOWN	0x0000
NWOT_USER	0x0001
NWOT_USER_GROUP	0x0002
NWOT_PRINT_QUEUE	0x0003
NWOT_FILE_SERVER	0x0004
NWOT_JOB_SERVER	0x0005
NWOT_GATEWAY	0x0006
NWOT_PRINT_SERVER	0x0007

NWOT_ARCHIVE_QUEUE	0x0008
NWOT_ARCHIVE_SERVER	0x0009
NWOT_JOB_QUEUE	0x000A
NWOT_ADMINISTRATION	0x000B
NWOT_NAS_SNA_GATEWAY	0x0021
NWOT_REMOTE_BRIDGE_SERVER	0x0024
NWOT_TIME_SYNCHRONIZATION_SERVER	0x002D
NWOT_ARCHIVE_SERVER_DYNAMIC_SAP	0x002E
NWOT_ADVERTISING_PRINT_SERVER	0x0047
NWOT_BTRIEVE_VAP	0x0050
NWOT_PRINT_QUEUE_USER	0x0053
NWOT_NVT_SERVER	0x009E

Bindery Property Types

NWBF_ITEM	0x00	Has only one value associated with it (such as PASSWORD)
NWBF_SET	0x02	Has many values associated with it (such as GROUPS_IM_IN)

Bindery Object and Property States

NWBF_STATIC	0x00	Permanent in bindery until deliberately deleted
NWBF_DYNAMIC	0x01	Is temporarily allocated and will be deleted when the server goes down

Bindery Object and Property Security Access Levels

NWBS_ANY_READ	0x00	Can be read by anyone
NWBS_LOGGED_READ	0x01	Must be logged in to read
NWBS_OBJECT_READ	0x02	Can be read by same object or supervisor
NWBS_SUPER_READ	0x03	Can be read only by supervisor
NWBS_BINDERY_READ	0x04	Can be read only by the bindery
NWBS_ANY_WRITE	0x00	Can be written by anyone
NWBS_LOGGED_WRITE	0x10	Must be logged in to write
NWBS_OBJECT_WRITE	0x20	Can be written by same object or supervisor
NWBS_SUPER_WRITE	0x30	Can be written only by the supervisor
NWBS_BINDERY_WRITE	0x40	Can be written only by the bindery

NWObjectInfo_t Structure

```
typedef struct {
        char    objectName[ NWMAX_OBJECT_NAME_LENGTH ];
        uint32  objectID;
        uint16  objectType;
        uint8   objectState;
        uint8   objectSecurity;
} NWObjectInfo_t;
```

objectName contains the name of the bindery object.

objectID is the unique ID that is assigned to all bindery objects.

objectType contains a bindery object type (see ªBindery Object Types.º)

objectState contains the state assigned to the object. (See ªBindery Object and Property States.º)

objectSecurity contains the security assigned to the object. (See ªBindery Object and Property Security Access Levels.º)

NWPropertyInfo_t Structure

```
typedef struct {
        char    propertyName[ NWMAX_PROPERTY_NAME_    LENGTH ];
        uint8   propertyStateAndType;
        uint8   propertySecurity;
        uint8   propertyHasAValue;
} NWPropertyInfo_t;
```

propertyName contains the name of the bindery property.

propertyStateAndType contains the state and type of the property. (See ªBindery Object and Property States.º) The Type flag is OR'ed together with the State flag.

propertySecurity contains the security assigned to the property. (See ªBindery Object and Property Security Access Levels.º)

propertyHasAValue contains 0 if there are no associated values or -1 (0xFF) if the property does have a value.

Information pertaining to Connection Services

Constant Definitions

NWMAX_CONNECTION_LIST_LENGTH	50
NWMAX_INTERNET_ADDRESS_LENGTH	12
NWMAX_KEYED_PASSWORD_LENGTH	8
NWMAX_LOGIN_TIME_LENGTH	7
NWMAX_OBJECT_NAME_LENGTH	48

Information pertaining to File and Path Services

Constant Definitions

NWMAX_DIR_NAME_LENGTH	16
NWMAX_DIR_PATH_LENGTH	255
NWMAX_DS_NAME	48
NWMAX_FILE_HANDLE_SIZE	6
NWMAX_FILE_NAME_LENGTH	16
NWMAX_NS_COUNT	10
NWMAX_NS_NAME	16
NWMAX_NUM_NS	10
NWMAX_NUM_DS	10
NWMAX_SERVER_NAME_LENGTH	48
NWMAX_USER_RESTRICTION	12
NWMAX_VOLUME_NAME_LENGTH	16

File Attributes

NWFA_NORMAL	0x00000000L
NWFA_READ_ONLY	0x00000001L
NWFA_HIDDEN	0x00000002L
NWFA_SYSTEM	0x00000004L
NWFA_EXECUTE_ONLY	0x00000008L
NWFA_NEED_ARCHIVE	0x00000020L
NWFA_SHARABLE	0x00000080L
NWFA_TRANSACTIONAL	0x00001000L
NWFA_INDEXED	0x00002000L
NWFA_READ_AUDIT	0x00004000L
NWFA_WRITE_AUDIT	0x00008000L
NWFA_PURGE	0x00010000L
NWFA_RENAME_INHIBIT	0x00020000L
NWFA_DELETE_INHIBIT	0x00040000L
NWFA_COPY_INHIBIT	0x00080000L

NWFA_NORMAL — The normal attribute allows read/write access to the file on both NetWare v2.*x* and NetWare v3.*x*.

NWFA_READ_ONLY — In NetWare v3.*x*, utilities such as FLAG automatically assign the delete inhibit and rename inhibit attributes with the read only attribute. In NetWare v2.*x*, the additional attributes are not set, but users cannot delete or rename the file until the read only attribute is removed.

NWFA_HIDDEN — In NetWare v2.*x* and v3.*x*, this attribute hides the file from DOS DIR scans and prevents it from being deleted or copied. The files will appear with NDIR.

NWFA_SYSTEM — In NetWare v2.*x* and v3.*x*, this attribute hides the file from DOS DIR scans and prevents it from being deleted or copies. However, the files will appear with NDIR.

NWFA_EXECUTE_ONLY — In NetWare v2.*x* and v3.*x*, this attribute prevents files from being copied. Only the supervisor can assign this attribute, and it should only be assigned if the file has been backed up. Once this bit is assigned, the bit cannot be deleted. This attribute can be set with FILER.

NWFA_NEED_ARCHIVE — In NetWare v2.*x* and v3.*x*, NetWare automatically assigns this bit to files that have been modified since the last backup.

NWFA_SHARABLE — In NetWare v2. *x* and v3.*x*, this attribute allows the file to be used by more than one user at a time and is usually used in combination with the read only attribute.

NWFA_TRANSACTIONAL

In NetWare v2. *x* and v3.*x*, this attribute indicates that files will be protected by TTS. TTS ensures that when a file is being modified either all changes are made or no changes are made.

NWFA_INDEXED — In NetWare v3.*x*, the indexed attribute is no longer supported since all files are automatically turbo FAT indexed when they have 64 or more regular FAT entries and are randomly accessed. However, this attribute can still be set or cleared for use with applications.

In NetWare v2.*x*, this attribute must be set for all files the user wants indexed. The operating system must also be configured for indexed files.

NWFA_READ_AUDIT	This attribute is associated with the NetWare Audit Trail System. The read and write audit files record information about who reads from and writes to a database file. Write audit makes continuous backup possible, and the combination of read and write audit provides greater security. This bit is currently not supported but will be in future releases.
NWFA_WRITE_AUDIT	See NWFA_READ_AUDIT. This bit is currently not supported but will be in future releases.
NWFA_PURGE	In NetWare v3.*x*, this attribute prevents the file from being salvageable. When assigned to a file, this attribute purges the file upon deletion. NetWare v2. *x* does not support this attribute.
NWFA_RENAME_INHIBIT	In NetWare v3.*x*, this attribute restricts users from renaming files even if they have the modify right. If they have the Modify right, they must remove this attribute before renaming. NetWare v2. *x* does not support this attribute.
NWFA_DELETE_INHIBIT	In NetWare v3.*x*, this attribute prevents users from erasing files even when they have been granted the Erase right at the file or directory level. If they have the Modify right, they can remove this attribute and then delete the file. NetWare v2. *x* does not support this attribute.
NWFA_COPY_INHIBIT	In NetWare v3.*x*, this attribute restricts only the copy rights of certain applications, such as the Macintosh Finder. If users have the Modify right, they can remove the copy inhibit attribute and then copy the file.

Directory Attributes

The file attributes listed below can be assigned to directories. However, in NetWare v2. *x*, only NWFA_HIDDEN and NWFA_SYSTEM can be assigned to directories.

NWFA_HIDDEN	0x00000002L
NWFA_SYSTEM	0x00000004L
NWFA_PURGE	0x00010000L
NWFA_RENAME_INHIBIT	0x00020000L
NWFA_DELETE_INHIBIT	0x00040000L

NWFA_HIDDEN	This directory attribute hides the directory from DOS DIR scans and prevents it from being deleted or copied. The directory will appear with NDIR.
NWFA_SYSTEM	This directory attribute hides the directory from DOS DIR scans and prevents it from being deleted or copied. The directory will appear with NDIR.
NWFA_PURGE	This directory attribute purges all files in the directory when the files are deleted. Such files cannot be recovered or salvaged.
NWFA_RENAME_INHIBIT	This directory attribute prevents users from renaming a directory even when they have been granted the Modify right. If they have the Modify right, they must remove this attribute before renaming the directory.
NWFA_DELETE_INHIBIT	This directory attribute prevents the users from deleting the directory even when they have been granted Erase rights to the directory. If they have the Modify right, they can remove this attribute and then delete the directory.

Search Attributes

NWSA_NONE	0x00	Normal files
NWSA_HIDDEN	0x02	Hidden and normal files
NWSA_SYSTEM	0x04	System and normal files
NWSA_BOTH	0x06	Hidden, system and normal files
NWSA_DIRECTORIES_ONLY	0x10	Normal directories only (cannot be used on v2.*x*)
NWSA_FILES_ONLY	0x20	Normal files only (cannot be used on v2.*x*)

A file (or directory) is designated system or hidden when its corresponding file (or directory) attribute is set. The Search Attributes are used to include system and/or hidden files (or directories) in a search. In other words, if only the system bit is set in the searchAttributes parameter then all files (or directories) will be affected except hidden files (or directories). If only the hidden bit is set, all files (or directories) will be affected except system files (or directories). When neither the hidden nor the system bit is set (0x00), then only files (or directories) that are not hidden, system, or both will be affected.

Trustee Rights and Inherited Rights Mask for NetWare v3.*x* and NetWare for UNIX software

The bit set by 0x0004 should be ignored by applications running on NetWare v3. *x* or NetWare for UNIX software. It is the "open bit" under v2. *x*.

Trustee Rights apply to both files and directories in NetWare v3. *x* and NetWare for UNIX software.

See "Trustee Access Rights and Maximum Rights Mask for NetWare v2. *x*" for an explanation of NetWare v2. *x* rights.

NWTR_NONE	0x0000
NWTR_READ	0x0001
NWTR_WRITE	0x0002
	0x0004 (Used only on NetWare v2. *x*)
NWTR_CREATE	0x0008
NWTR_ERASE	0x0010
NWTR_ACCESS	0x0020
NWTR_FILE_SCAN	0x0040
NWTR_MODIFY	0x0080
NWTR_SUPERVISOR	0x0100
NWTR_NORMAL	0x00FF
NWTR_ALL	0x01FF

Each right is described below.

NWTR_READ Directories: User can open and read existing files unless blocked by mask or trustee rights assignments.

Files: User can open and read this file.

NWTR_WRITE Directories: User can open and write to files in this directory unless blocked by mask or trustee rights assignment.

Files: User can open and write to this file.

NWTR_CREATE Directories: User can create files and subdirectories in this directory.

Files: User can salvage this file if it is deleted.

NWTR_ERASE Directories: User can delete this directory if the user has rights to delete everything inside it.

	Files: User can delete this file
NWTR_ACCESS	Directories: User can modify the trustee list and Inherited Rights Mask of this directory.
	Files: User can modify this file's trustee list and Inherited Rights Mask.
NWTR_FILE_SCAN	Directories: When scanning the directory, user can see the names of files in this directory unless blocked by mask or trustee rights assignment.
	Files: When scanning the directory, user can see the name of this file.
NWTR_MODIFY	Directories: User can rename this directory and change the attributes of it.
	Files: User can rename the file and change its attributes.
NWTR_SUPERVISOR	Directories: User has all rights to this directory and all subdirectories and files. User can grant supervisor rights to other users in this directory and in subdirectories and files. User's rights override all inherited rights masks in subdirectories and files. User can assign space limitations to subdirectories.
	Files: User has all rights to this file.
NWTR_NORMAL	Directories: User has all of the above rights in this directory, except NWTR_SUPERVISOR.
	Files: User has all of the above rights to this file, except NWTR_SUPERVISOR.
NWTR_ALL	Directories: User has all of the above rights in this directory.
	Files: User has all of the above rights to this file.

Trustee Access Rights and Maximum Rights Mask for NetWare v2.*x*

NetWare v2.*x* has the following rights.

NWTA_NONE	0x00
NWTA_READ	0x01
NWTA_WRITE	0x02
NWTA_OPEN	0x04
NWTA_CREATE	0x08
NWTA_DELETE	0x10
NWTA_OWNERSHIP	0x20
NWTA_SEARCH	0x40
NWTA_MODIFY	0x80
NWTA_ALL	0xFF

Each right is defined below.

NWTA_READ	If the user also has the NWTA_OPEN right, the user can open and read existing files in this directory unless blocked by the maximum rights mask.
NWTA_WRITE	If the user also has the NWTA_WRITE right, the user can open and write to files in this directory unless blocked by the maximum rights mask.
NWTA_OPEN	The user can open existing files. If the user also has NWTA_WRITE and NWTA_READ rights, the user can view and change the contents of such files.
NWTA_CREATE	The user can create and salvage files in this directory unless blocked by the

maximum rights mask. The Ownership right along with the Create right is required to create directories.

NWTA_DELETE — The user can delete files. The Ownership right along with the Delete right is required to delete directories.

NWTA_OWNERSHIP — The user can create, rename, or delete subdirectories of the directory if the user also has the additional needed right: Create to create, Modify to rename, or Delete to delete. The user can also modify the trustee list and maximum rights mask of this directory and its subdirectories. In the NetWare utilities, this is the Parental right.

NWTA_SEARCH — The user can see the names of files and subdirec tories in this directory unless blocked by the maximum rights mask.

NWTA_MODIFY — The user can also change the attributes of the directory, its files and subdirectories. If the user also has the Ownership right, the user can rename the files and subdirectories.

NWTA_ALL — The user has all rights in this directory.

Change Attributes

Change attributes are used with the NWSetDirEntryInfo or NWSetFileEntryInfo function calls. These values can be OR'ed together. In these functions, you pass a structure containing the new values you want to set, and then use the following change attributes to specify which fields within the structure contain the new values.

NWCA_NAME	0x0001
NWCA_ATTRIBUTES	0x0002
NWCA_CREATE_DATE_AND_TIME	0x000C
NWCA_OWNER_ID	0x0010
NWCA_LAST_ARCHIVED_DATE_AND_TIME	0x0060
NWCA_LAST_ARCHIVED_ID	0x0080
NWCA_LAST_MODIFY_DATE_AND_TIME	0x0300
NWCA_LAST_MODIFY_ID	0x0400
NWCA_LAST_ACCESSED_DATE	0x0800
NWCA_INHERITED_RIGHTS_MASK	0x1000
NWCA_DIR_RESTRICTION	0x2000

Note: NWCA_LAST_ACCESSED_DATE and NWCA_LAST_MODIFY_ID are not available for use with directories.

NWCA_NAME — Changes the name of the directory or file.

NWCA_ATTRIBUTES — Changes the attributes of the directory or file. See "File Attributes" and "Directory Attributes." Users must have Modify rights to the file or directory to change attributes.

NWCA_CREATE_DATE_AND_TIME — Changes the date and time the file or directory was created. Users must have supervisor equivalence to change the date and time.

NWCA_OWNER_ID — Changes the owner of the file or directory by changing the object ID in the field. Users must have supervisor equivalence to change the owner of a file or directory.

NWCA_LAST_ARCHIVED_DATE_AND_TIME — Changes the date and time the file or directory was archived.

NWCA_LAST_ARCHIVED_ID — Changes the object ID to the user who archived the file or directory.

NWCA_LAST_MODIFY_DATE_AND_TIME
Changes the date and time the file or directory was modified. NetWare automatically updates this information when a file or directory is modified. The user must be supervisor equivalent to change this information with an API.

NWCA_LAST_MODIFY_ID Changes the object ID to the user who modified the file or directory. NetWare automatically updates this field when a directory or file is changed. The user must be supervisor equivalent to change this information with an API.

NWCA_LAST_ACCESSED_DATE
Changes the date and time the file was accessed. NetWare automatically updates this information when a file or directory is accessed. The user must be supervisor equivalent to change this information with an API.

NWCA_INHERITED_RIGHTS_MASK
Changes the trustee rights in the file's or directory's Inherited Rights Mask. See "Trustee Rights and Inherited Rights Mask for NetWare v3. *x* and NetWare for UNIX" for a list of rights that can be passed. The user must have access control rights to the file or directory to change the inherited rights mask.

NWCA_DIR_RESTRICTION Changes the directory restrictions. A 0 clears all restrictions; a 1 restricts the directory to 4KB; a 2, to 8KB; etc.

"Open file" Access Rights

These definitions are used with NWOpenFile. One or both of the following rights must be assigned to the accessRights parameter:

NWOR_READ	0x01	Opens the file for reading.
NWOR_WRITE	0x02	Opens the file for writing.

The above NWOR_READ or NWOR_WRITE bit mask may be OR'ed with either NWOR_DENY_READ and NWOR_DENY_WRITE or NWOR_COMPATIBILITY.

NWOR_DENY_READ	0x04	Doesn't allow others to read fromthe file while you have it open
NWOR_DENY_WRITE	0x08	Doesn't allow others to write to the file while you have it open
NWOR_COMPATIBILITY	0x10	Determines access in connection with the file's "sharable" attribute.

When the NWOR_COMPATIBILITY bit is set, the following things apply:

- NWOR_DENY_READ and NWOR_DENY_WRITE are not applicable, because they will be ignored.
- When the sharable file attribute is set, the file is treated as a sharable file, no user having exclusive access.
- When the sharable file attribute is not set, one of the following occurs:
  - If NWOR_COMPATIBILITY bit is OR'ed with NWOR_READ, the file is opened with write access denied to other users.
  - If NWOR_COMPATIBILITY bit is OR'ed with NWOR_WRITE or both NWOR_READ and NWOR_WRITE, the file is opened with read and write access denied to other users.

All of the above information is only applicable if the open call is successful.

The following access right is available for NetWare v3.*x* and may be OR'ed with any of the above values:

NWOR_SYNC_MODE	0x40	Allows "write-through" access; that is, writes directly to the disk, bypassing any caching and/or buffering. NetWare for UNIX software does not support this flag.

Directory and File Handle definitions

```
typedef  uint8     NWDirHandle_ts;
typedef  uint8     NWFileHandle_ta[NWMAX_FILE_HANDLE_SIZE];
```

Handles are values that represent a complete path and file (or directory) name as defined in the file server's file handle or directory handle index table. These handles can be used to specify a file or directory without passing a complete path name. But in order to use them, you must keep track of them since there are no NWGetHandle functions.

NWPath_t Structure

```
typedef struct {
        NWDirHandle_ts      dirHandle;
        uint16              serverConnID;
        char                *pathName;
} NWPath_t;
```

The dirHandle field contains either of the following:

- A 0, when a full path will be given in the pathName field.
- A value representing an allocated directory Handle.

See NWAllocTemporaryDirHandle or NWAllocPermanentDirHandle.

serverConnID is the value corresponding to the server attachment. (See ªNWAttachToServerPlatform.º)

pathName contains the address of a path that is either 1) a full path when a 0 dirHandle is used, or 2) a relative path when an allocated dirHandle is used. (The path is relative to the directory that the dirHandle represents.)

Note: No space is allocated for pathName. The application must allocate space for the path separately and then assign pathName the address of the previously allocated space.

NWDirEntryInfo_t Structure

```
typedef struct {
        uint32    attributes;
        uint32    creationDateAndTime;
        uint32    ownerID;
        uint32    archiveDateAndTime;
        uint32    archiverID;
        uint32    lastModifyDateAndTime;
        uint32    dirRestriction;
        uint16    inheritedRightsMask;
        uint8     nameSpaceID;
        char      entryName[ NWMAX_DIR_NAME_LENGTH ];
} NWDirEntryInfo_t;
```

To change these fields with NWSetDirEntryInfo, you must use the Change Attributes.

attributes for directories contains a bit mask of the directory's attributes. This value will be zero when using

NWScanDirEntryInfo on a file server running NetWare v2.*x*.

creationDateAndTime contains the date and time the directory was created.

ownerID is the object ID of the directory's owner. You can use NWGetObjectName to get the name of the object.

archiveDateAndTime contains the date and time the directory was last archived.

archiverID contains the objectID of the object that archived the directory. You can use NWGetObjectName to get the name of the object.

lastModifyDateAndTime contains the date and time the directory was last modified.

dirRestriction contains the number of 4K blocks available to that directory and its subdirectories. This field is set with NWSetDirRestriction. The default is 0.

inheritedRightsMask represents the inherited rights mask of the current directory. See ªTrustee Rights and Inherited Rights Mask for NetWare v3. *x* and NetWare for UNIX.º

nameSpaceID contains a 0 if the directory is a DOS directory and 1 if the directory is a Macintosh directory, or other numbers representing other name spaces, if the file server has been configured for them.

entryName contains the directory name.

NWFileEntryInfo_t Structure

```
typedef struct {
        uint32  attributes;
        uint32  creationDateAndTime;
        uint32  ownerID;
        uint32  archiveDateAndTime;
        uint32  archiverID;
        uint32  updateDateAndTime;
        uint32  updatorID;
        uint32  fileSize;
        uint32  lastAccessDateAndTime;
        uint16  inheritedRightsMask;
        uint8   nameSpaceID;
        char    entryName[ NWMAX_FILE_NAME_LENGTH ];
} NWFileEntryInfo_t;
```

To change these fields with NWSetFileEntryInfo, you must use the Change Attributes.

attributes for the NWFileEntryInfo structure contains the current file's attributes.

creationDateAndTime contains the date and time the file was created. The time will always be 0 for NetWare v2.*x*.

ownerID is the object ID of the file's owner. You can use NWGetObjectName to get the name of the object.

archiveDateAndTime contains the date and time the file was last archived.

archiverID contains the objectID of the object that archived the file. You can use NWGetObjectName to get the name of the object.

updateDateAndTime contains the objectID of the object that updated the file.

updatorID contains the objectID of the object that updated the file. You can use NWGetObjectName to get the name of the object.

fileSize contains the file size in bytes.

lastAccessDateAndTime contains the date and time when the file was last accessed. The time will always be 0.

inheritedRightsMask represents the inherited rights mask of the file. See "Trustee Rights and Inherited Rights Mask for NetWare v3. *x* and NetWare for UNIX" on page A-10.

nameSpaceID contains a 0 if the directory is a DOS directory and 1 if the directory is a Macintosh directory, or other numbers representing other name spaces, if the file server has been configured for them.

entryName contains the file name.

NWVolUsage_t Structure

```
typedef struct {
        uint32   totalBlocks;
        uint32   availableBlocks;
        uint32   purgableBlocks;
        uint32   notYetPurgableBlocks;
        uint32   totalDirEntries;
        uint32   availDirEntries;
        uint32   maxDirEntriesUsed;
        uint16   volNum;
        uint16   sectorsPerBlock;
        uint8    isHashed;
        uint8    isCached;
        uint8    isRemovable;
        uint8    isMounted;
        char     volName[ NWMAX_VOLUME_NAME_LENGTH ];
} NWVolUsage_t;
```

This structure contains some fields that are only pertinent to NetWare v2.*x* and some that are only pertinent to NetWare v3.*x* and NetWare for UNIX software.

totalBlocks contains the total amount of 4K blocks allocated to the volume.
NetWare for UNIX returns the total amount of disk space on the host system. All volumes mounted from the same file system will return the same value.

availableBlocks contains the amount of 4K blocks not used. NetWare for UNIX returns the total amount of disk space on the host system. All volumes mounted from the same file system will return the same value.

purgableBlocks contains the amount of blocks marked for deletion and purgeable. A valid value is only returned from servers running NetWare v3.*x*.
NetWare for UNIX does not support this field and returns a 0.

notYetPurgableBlocks contains the amount of blocks marked for deletion, but not yet purgeable, because the file server holds deleted files for a certain amount of time, before allowing them to be purged (as set by the minimum file delete wait time console command). A valid value is only returned from servers running NetWare v3.*x*.
NetWare for UNIX does not support this field and returns a 0.

totalDirEntries contains the total amount of directories which can be created. NetWare for UNIX returns the number of files that are available to keep track of NetWare file and trustee information.

availDirEntries contains the amount of directories which can be created, based on the difference between the total amount of directories and the amount of directories already created. NetWare for UNIX returns the number of

directories that can be created.

maxDirEntriesUsed contains the maximum number of directories in use at one time since the volume was created. NetWare for UNIX does not support this field and returns a 0.

volNum contains the number assigned by the file server to each volume name (beginning with 0). NetWare for UNIX returns a volume number according to the order the volumes are listed in the NWConfig file.

sectorsPerBlock contains the number of 512 sectors per block within a volume. For v3.*x* servers this number will always be 8. For v2.*x* servers this is configurable from 1-16. NetWare for UNIX does not support this field and returns a 0.

isHashed contains a 0 if volume entries are not hashed, and non-zero if the volume entries are hashed. A valid value is only returned from servers running
NetWare v2.*x*.

isCached contains 0 if volume entries are not cached and non-zero if the volume entries are cached. A valid value is only returned from servers running NetWare v2.*x*.

*isRemovable*l contains 0 if the volume is on fixed media, and a non-zero value if the volume is on a removable (mountable) medium. NetWare for UNIX does not support this field and returns a 0.

isMounted contains a 0 if the volume is not mounted and non-zero otherwise. A valid value is only returned from servers running NetWare v2.*x*.

volName contains the name of the volume. Maximum length is 16 characters. NetWare for UNIX returns the name as defined in the NWConfig file.

NWDirRestriction_t Structure

```
typedef struct {
        uint16    level;
        uint32    maxBlocks;
        uint32    availableBlocks;
} NWDirRestriction_t;
```

level refers to the distance from the directory to the root directory.

maxBlocks contains the maximum amount of space assigned to the directory. Blocks are in 4K units.

All directories will have a value in the maxBlocks parameter. The maxBlocks parameter will return one of the following:

0x7FFFFFFF	No restrictions have ever been set.
negative value	Restrictions were set but they have been cleared. (Use a zero in NWSetDirRestriction to clear restrictions.)
positive value	Restrictions are set, and the positive value is the maximum value.

availableBlocks contains the amount of space assigned to a directory minus the amount of space used by the directory and its subdirectories. Blocks are in 4K units.

To calculate the amount of space in use, simply subtract availableBlocks from maxBlocks.

NWUserRestriction_t Structure

```
typedef struct {
        uint32    objectID;
```

```
        uint32    restriction;
} NWUserRestriction_t
```

objectID contains the bindery object ID of the object corresponding to the restriction.

restriction is specified in 4K blocks and represents the space restrictions placed within a volume on a particular object.

NWTrusteeRights_t Structure

```
typedef struct {
        uint32    trusteeID;
        uint16    trusteeRights;
} NWTrusteeRights_t;
```

trusteeID contains the bindery object ID of the trustee.

trusteeRights refers to the rights given to the trustee for a particular directory (or file in NetWare 3.*x*). (See ªTrustee Rights and Inherited Rights Mask.º)

NWSalvageableInfo_t Structure

```
typedef struct {
        uint32              deletedDateAndTime;
        uint32              deleterID;
        uint32              attributes;
        uint32              creationDateAndTime;
        uint32              ownerID;
        uint32              archiveDateAndTime;
        uint32              archiverID;
        uint32              updateDateAndTime;
        uint32              updatorID;
        uint32    fileSize;
        uint32              inheritedRightsMask;
        uint32              lastAccessDateAndTime;
        uint8               nameSpaceID;
        char                fileName[NWMAX_FILE_NAME_LENGTH];
} NWSalvageableInfo_t;
```

deletedDateAndTime contains the date and time that the file was deleted.

deletedID refers to the bindery object ID of the object that deleted the file.

attributes contains a value representing the file's set attributes.

creationDateAndTime contains the date and time that the file was created.

ownerID refers to the bindery object ID of the file's owner. You can use NWGetObjectName to get the name of the object.

archiveDateAndTime contains the date and time that the file was last archived.

archiverID contains the bindery object ID of the object that archived the file. You can use NWGetObjectName to get the name of the object.

updateDateAndTime contains the date and time that the file was last changed.

updatorID refers to the bindery object ID of the object that made changes to the file. You can use NWGetObjectName to get the name of the object.

fileSize contains the size of the file.

inheritedRightsMask contains a value representing the inherited rights owned by the file.

lastAccessDateAndTime contains the date and time that the file was last accessed.

nameSpaceID contains the name space for the file (0 for DOS, 1 for Macintosh).

fileName contains the name of the file.

NWDataStreamInfo_t Structure

```
typedef struct {
        uint8                   associatedNameSpace;
        char                    dataStreamName[NWMAX_DS_NAME];
} NWDataStreamInfo_t;
```

associatedNameSpace contains the ID number of the name space associated with this data stream.

dataStreamName contains the name of the data stream.

NWNameSpaceInfo_t Structure

```
typedef struct {
        uint8                   definedNameSpaces;
        char                    nameSpaceName[NWMAX_NUM_NS]
[NWMAX_NS_NAME];
        uint8                   definedDataStreams;
        NWDataStreamInfo_t dataStream[NWMAX_NUM_DS];
        uint8                   loadedNSCount;
        uint8                   loadedNS[NWMAX_NS_COUNT];
        uint8                   volumesNSCount;
        uint8                   volumesNS[NWMAX_NS_COUNT];
        uint8                   volumesDSCount;
        uint8                   volumesDS[NWMAX_NS_COUNT];

} NWNameSpaceInfo_t;
```

definedNameSpaces contains the number of name spaces defined on the file server.

nameSpaceName contains the names of the name spaces that the file server supports.

definedDataStreams contains the number of data streams the file server has been configured for.

dataStream contains a pointer to the NWDataStreamInfo_t Structure which contains the ID number of the associated name space and the data stream name.

loadedNSCount contains the number of name spaces actually loaded on the file server.

loadedNS contains an index into the defined name space table.

volumesNSCount contains the number of name spaces that the volume is using.

volumesNS contains an index into the defined name space table.

volumesDSCount contains the number of data streams that the volume is using.

volumesDS contains an index into the defined data stream table.

Information pertaining to Queue Management Services

Constant Definitions

NWMAX_BANNER_NAME_FIELD_LENGTH	13
NWMAX_BANNER_FILE_FIELD_LENGTH	13
NWMAX_CLIENT_RECORD_LENGTH	152
NWMAX_FORM_NAME_LENGTH	16
NWMAX_HEADER_FILE_NAME_LENGTH	14
NWMAX_JOB_DESCRIPTION_LENGTH	50
NWMAX_JOB_DIR_PATH_LENGTH	80
NWMAX_JOB_FILE_NAME_LENGTH	14
NWMAX_JOB_STRUCT_SIZE	256
NWMAX_NUMBER_OF_JOB_NUMBERS	250
NWMAX_NUMBER_OF_SERVER_CONN_NUMBERS	25
NWMAX_NUMBER_OF_SERVER_OBJECT_IDS	25
NWMAX_QUEUE_JOB_TIME_SIZE	6
NWMAX_QUEUE_NAME_LENGTH	48
NWMAX_QUEUE_SUBDIR_LENGTH	119
NWMAX_SERVER_STATUS_RECORD_LENGTH	64

Queue Status Flags

NWQS_NO_SERVER_RESTRICTIONS	0x00
NWQS_NO_MORE_JOBS	0x01
NWQS_NO_MORE_SERVER_ATTACHMENTS	0x02
NWQS_SERVERS_DISABLED	0x04

NWQueueJobStruct_t Structure

```
typedef struct {
        uint8               clientStation;
        uint8               clientTask;
        uint32              clientID;
        uint32              targetServerID;
        uint8               targetExecutionTime[NWMAX_QUEUE_
                                    JOB_TIME_SIZE];
        uint8               jobEntryTime[NWMAX_QUEUE_JOB_TIME_SIZE];
        uint16              jobNumber;
        uint16              jobType;
        uint8               jobPosition;
        uint8               jobControlFlags;
        uint8               jobFileName[NWMAX_JOB_FILE_NAME_
                                    LENGTH];
        NWFileHandle_ta     jobFileHandle;
        uint8               servicingServerStation;
        uint8               servicingServerTaskNumber;
        uint32              servicingServerIDNumber;
        uint8               jobDescription[NWMAX_JOB_DESCRIPTION_
                                    LENGTH];
        NWClientRecord_ta   queueRecord;
} NWQueueJobStruct_t;

typedef   char   NWClientRecord_ta[NWMAX_CLIENT_RECORD_LENGTH];
```

Of the fields defined in the NWQueueJobStruct_t structure, the user can modify only those described below.

targetServerID contains the server ID of the queue server that will service the job. If this field is set to 0xFFFFFFFF, any queue server can service the job. If the specified queue server is not attached to the queue, QMS removes the job from the queue.

targetExecutionTime indicates the earliest time that the job can be serviced. The bytes are assigned as follows: year, month, day, hour, minute, second. If this field is set to 0xFFFFFFFFFFFF, the job will be serviced at the first opportunity.

jobType contains a number that identifies the type of job entry. A queue server can request specific job types from a queue.

jobControlFlags contains flag bits indicating the status of the job. Bits in the field are set as follows:

NWCF_OPERATOR_HOLD	0x80
NWCF_USER_HOLD	0x40
NWCF_ENTRY_OPEN	0x20
NWCF_SERVICE_RESTART	0x10
NWCF_SERVICE_AUTO_START	0x08

- When the NWCF_SERVICE_AUTO_START is set, the job will be serviced after a queue server connection is broken, even if the client has not cleared the Entry Open bit. If the bit is cleared when a server connection is broken, QMS removes the job from the queue.
- When the NWCF_SERVICE_RESTART is set, the job remains in the queue (in its current position) when a queue server fails. If this bit is cleared, QMS removes the job from the queue when a server fails.
- When the NWCF_ENTRY_OPEN is set, the client has not filled the associated job file. The NWCloseFileAndStartQueueJob function clears this bit, marking the job is ready for service, if the User Hold and Operator Hold bits are cleared.
- When the NWCF_USER_HOLD is set, the job continues to advance in the queue, but cannot be serviced until a client or operator clears this bit.
- When the NWCF_OPERATOR_HOLD is set, the job continues to advance in the queue, but cannot be serviced until the operator clears this bit.

jobDescription contains a null-terminated ASCII text description of the content or purpose of a job. QMS displays this text as part of the job description when users or operators examine a queue.

queueRecord may contain up to 152 bytes. This field is application dependent, and its format is entirely application dependent. The NetWare print server uses a NWPrintRecord_t Structure as the format for this field.

NWPrintRecord_t Structure

```
typedef struct {
        uint8       versionNumber;
        uint8       tabSize;
        uint16      numCopies;
        uint16      controlFlags;
        uint16      linesPerPage;
        uint16      charsPerLine;
        char        formName[NWMAX_FORM_NAME_LENGTH];
        char        bannerNameField[NWMAX_BANNER_
                        NAME_FIELD_LENGTH];
        char        bannerFileField[NWMAX_BANNER_
                        FILE_FIELD_LENGTH];
        char        headerFileName[NWMAX_HEADER_FILE_
                        NAME_LENGTH];
        char        directoryPath[NWMAX_JOB_DIR_PATH_LENGTH];
```

} NWPrintRecord_t;

versionNumber currently contains 0.

tabSize contains the number of spaces tabs will be expanded to (0-18).

numCopies contains the number of copies that will be printed.

controlFlags contains one or more of the following:

NWPCF_SUPPRESS_FF	0x0008
NWPCF_NOTIFY_USER	0x0010
NWPCF_TEXT_MODE	0x0040
NWPCF_PRINT_BANNER	0x0080

- When NWPCF_SUPPRESS_FF is set, the form feed is suppressed.
- When NWPCF_NOTIFY_USER is set, the user is notified that the job is finished.
- When NWPCF_TEXT_MODE is set, tabs are expanded and the lines per page and characters per line are ignored.
- When NWPCF_PRINT_BANNER is set, a banner is printed.

linesPerPage refers to the number of lines on one page. The design default is 66, but a default value is not currently implemented.

charsPerLine contains the number of characters on one line. The design default is 132, but a default value is not currently implemented.

formName contains the name of the form to be used in printing.

bannerNameField contains the text that is printed in first box of banner-usually used for user name.

bannerFileField contains the text printed in second box in banner-usually used for file name.

headerFileName contains the file name that is printed in header of banner.

directoryPath contains the full path name of directory, where the file resides.

Information pertaining to Server Platform Services

Constant Definitions

NWMAX_CONNECTION_LIST_LENGTH	50
NWMAX_COMPANY_NAME_LENGTH	80
NWMAX_COPYRIGHT_NOTICE_LENGTH	80
NWMAX_DATE_LENGTH	24
NWMAX_DESCRIPTION_LENGTH	80
NWMAX_OBJECT_NAME_LENGTH	48
NWMAX_SERVER_NAME_LENGTH	48

NWDescriptionStrings_t Structure

```
typedef struct {
        char    companyName[ NWMAX_COMPANY_NAME_LENGTH ];
        char    revisionDescription[ NWMAX_DESCRIPTION_LENGTH ];
        char    revisionDate[ NWMAX_DATE_LENGTH ];
        char    copyrightNotice[ NWMAX_COPYRIGHT_NOTICE_    LENGTH ];
} NWDescriptionStrings_t;
```

Each field in this structure contains a null termination.

Note: For applications talking to NetWare for UNIX servers, these strings may differ.

companyName receives the name of the company that is providing this version of NetWare.

revisionDescription receives the NetWare version and revision description string.

revisionDate receives the revision date in the form 02/15/1988.

copyrightNotice passes a pointer to the string allocated for the copyright notice.

NWServerPlatformDateAndTime_t Structure

```
typedef struct {
        uint8           year;
        uint8           month;
        uint8           day;
        uint8           hour;
        uint8           minute;
        uint8           second;
        uint8           dayOfWeek;
} NWServerPlatformDateAndTime_t;
```

The date and time are passed in with the following values:

year	becomes	0 through 99; for example: 82=1982
month	becomes	1 through 12
day	becomes	1 through 31
hour	becomes	0 through 23
minute	becomes	0 through 59
second	becomes	0 through 59
dayOfWeek	becomes	0 through 6 with 0 being Sunday

NWServerPlatformInfo_t Structure

```
typedef struct {
        uint16          majorVersion;
        uint16          minorVersion;
        uint16          revision;
        uint16          SFTLevel;
        uint16          TTSLevel;
        uint16          accountingVersion;
        uint16          VAPVersion;
        uint16          queueingVersion;
        uint16          printServerVersion;
        uint16          virtualConsoleVersion;
        uint16          securityRestrictionLevel;
        uint16          internetBridgeSupport;
        uint16          maxClientConnSupported;
        uint16          clientConnInUse;
        uint16          peakClientConnUsed;
        uint16          maxVolumes;
        char            serverName[ NWMAX_SERVER_NAME_LENGTH ];
} NWServerPlatformInfo_t;
```

majorVersion contains the major NetWare version number.

minorVersion contains the minor version (or subVersion) number.

revision refers to the revision level of the NetWare version number.

SFTLevel indicates which SFT level the file server operating system is using.

TTSLevel indicates which TTS level the file server operating system is using.

accountingVersion contains the accounting version number.

VAPVersion contains the VAP version number.

queueingVersion refers to the queuing version number.

printServerVersion contains the print server version number.

virtualConsoleVersion contains the virtual console version number.

securityRestrictionLevel contains the security restriction version number.

internetBridgeSupport contains the internet bridge support version number.

maxClientConnSupported indicates the maximum number of connections the file server can support.

clientConnInUse contains the number of connections that are currently using the file server.

peakClientConnUsed indicates the maximum number of connections in use at one time.

maxVolumes contains the maximum allowable number of volumes. For NetWare v3.*x*, the maximum is 32. For NetWare for UNIX, the maximum is configurable.

serverName contains the name of the server platform.

Information pertaining to the Synchronization Services

Constant Definitions

NWMAX_LOGICAL_RECORD_NAME_LENGTH	80
NWMAX_SEMAPHORE_NAME_LENGTH	127

File Log Flags

For use with log file calls.

NWFL_LOG_ONLY	0x00
NWFL_LOG_AND_LOCK	0x01

Record Log Flags

For use with physical and logical record log calls.

NWPL_LOG_ONLY	0x00
NWPL_LOG_AND_LOCK_EXCLUSIVE	0x01
NWPL_LOG_AND_LOCK_SHAREABLE	0x03

If the low-order bit is off, then the file is only logged. If the low-order bit is on, then the file is logged and locked. The high-order bit determines whether the file is locked exclusive or locked shareable. Locked has a value of 1; exclusive, 0; and shareable, 2. Thus locked exclusive is 0x01, and locked shareable is 0x03.

Record Lock Set Flags

For use with physical and logical record lock set calls.

NWLS_EXCLUSIVE	0x00
NWLS_SHAREABLE	0x02