avanti technology, inc.

TaskMaster® v5 for NetWare®

User`s Guide

(v5.04 - May 16 2011)

Table of Contents

Chapter 1 - Installation

TaskMaster® can be used to automate virtually any task normally run on the Server, as well as many file management tasks which typically require operator interaction. Once loaded on a Server, the Server module:

- Adds a secondary Console (or Shell, the TaskMaster Console {TMConsole}) extending the basic Server System Console command set with numerous additional commands. The TMConsole (Shell) provides so many Server and File System management commands that it is often referred to as a DOS Console (Shell) for the Server. With the Full TaskMaster version (i.e., not TaskMaster Lite {TMLite}), Server-to-Server data transfer and task management operations are also supported via the TMConsole (Shell) extended Console Commands.

- Adds an enhanced Batch Processor and extended Scripting Language for automating tasks. TaskMaster's Batch Processor can process basic NetWare Command Files (.NCF) with their limited System Console commands, plus TaskMaster provides which include TMConsole's extended Console Commands, IF/WHILE Conditional Tests / Structures, extended Batch Commands, Dynamic Variables, and Environment Variables. Using these features, it is possible to create sophisticated task scripts which can automate virtually any Server task.

- Adds a flexible Scheduler allowing both basic NetWare Command Files (.NCF) and enhanced TaskMaster Task files (.TSK) to be scheduled as needed, including support for multiple concurrent task execution.

- Adds Bindery/NDS Secure Remote Console support (via TaskMaster's Remote Console DOS Client) with additional configuration options allowing separate classes of access, including secure view only access. (Note: *Full TaskMaster version only, not supported by TaskMaster Lite {TMLITE}*).

Utilizing TaskMaster's capabilities, Server resources can be managed on a more automated and proactive basis than possible using the basic options and tools included in the Server OS. TaskMaster is available in two versions:

TaskMaster
This full featured Server Module supports not only the automation of tasks on the Local Server but the ability to interact with Remote Servers via additional Console Command Extensions, including the SYNC Console Command Extension which simplifies the replication / synchronization of data / files across directories, volumes, and Servers. In addition, the accompanying Remote Console DOS Client provides a more robust and secure method for Server Console access than Novell's RCONSOLE Client, as well as the ability to submit commands to the Server Console or tasks to TaskMaster in either a command-line or batch execution mode. This combination of features makes TaskMaster not only an exceptional Server management solution but a powerful network management tool.

TaskMaster Lite (TMLite)
An entry level, Server-centric version of the TaskMaster Server Module which has the following limitations:

- Remote Server (Server-to-Server) Console Command extensions / operations are not supported;
- SYNC (Console Command extension used for data replication / synchronization) is not supported;
- DOS Client support is not included (i.e., no Remote Console access or Client submission of commands / tasks).

The features in TMLite make it an exceptional Server management solution and an alternative for single Server installations or networks where Remote Server or network interaction is not a benefit for Server task automation.

Installing the Software

Before installing the software, the following checklists should be determined:

General Installation Checklist:
- Destination for the files (UNC: \\Server\Vol\path)
- Physical drive mapped to the same volume as the Server files destination
- Logged in to the Server as SUPERVISOR (Bindery), Admin (NDS), or equivalent

Novell NetWare Server Module Checklist:
- Refer to the Technical Support chapter for compatibility requirements

Novell NetWare NetWare Client Checklist:
- Client 32 for NetWare fully installed and supporting client access to NetWare Servers

Task Management Windows Client Checklist:
- OS support for Microsoft Windows 32-bit (Windows 2000 / NT / XP / Vista) executables

Remote Console DOS Client Checklist:
- OS support for Microsoft MS-DOS v3 (or later) executables (including Windows DOS-box execution)

The software is distributed using a self-extracting archive program which will prompt for the destination location to extract and install the archived files. The location specified should be a secure directory on the Server where the Server Module is intended to be loaded.

Note: If the intended destination directory does not already exist, the program will prompt for permission to attempt to create it.

Once the archived files have been extracted and installed on a single Server, it is typically easier to copy the files to other Servers where installation is intended. The self-extracting archive program is merely a simplified means for insuring the integrity of the distributed files and automating their initial installation. It does not modify the Workstation's registry nor does it alter the Server's registry or Bindery / NDS / eDirectory schemas. The self-extracting archive contains the following distribution files:

TaskMaster (Full Version)

README	(Notes pertinent to the release)
MANUAL.PDF	(Adobe Acrobat format User's Guide)
QUICKREF.PDF	(Adobe Acrobat format Quick Reference)
TASKMSTR.NLM	(Master Server module)
TMCLIENT.EXE	(Task Management Windows Client)
TMCMDHLP.DAT	(TMConsole Help)
TMREMOTE.EXE	(Remote Console DOS Client)
TMSECURE.NLM	(Secure Server module)

TaskMaster Lite (TMLite)

README	(Notes pertinent to the release)
QUICKREF.PDF	(Adobe Acrobat format Quick Reference)
MANUAL.PDF	(Adobe Acrobat format User's Guide)
TMLITE.NLM	(Server Module)
TMCLIENT.EXE	(Task Management Windows Client)
TMCMDHLP.DAT	(TMConsole Help)

Chapter 2 - Getting Started

TaskMaster® consists of two sets of components: A Remote Console DOS Client and two Server-based modules, each designed for a specific purpose. **TaskMaster Lite (TMLite®)** consists of only a single Server-based module.

Server Module

The Server Module (NLM) is the only required component for automating tasks since it facilitates the Console Command extensions, the enhanced Batch Processor, and the flexible Task Scheduler. . . All in One NLM! Unlike other alternatives which require the orchestration of a conglomeration of NLMs and support files in order to automate a task, TaskMaster's single NLM Server Module can be used to automate virtually any Server task.

TaskMaster

The Server module also provides secure Remote Server Console access to the Remote Console DOS Client. When loaded on multiple Servers in a network, the Server modules are capable of performing Server-to-Server operations. Some of the operations that can be performed: Copying files between Servers, submitting commands to Remote Servers, and launching tasks on Remote Servers.

There are two Server modules:

- Master Server module (TASKMSTR.NLM)
- Secure Server module (TMSECURE.NLM)

Either of these can be loaded, but not both concurrently.

Selecting the Appropriate Server Module

The Master Server module is more robust and full featured. In addition to supporting all the batch processing, task scheduling, and remote access functions, it also extends the Server's basic System Console Command set. The extended commands support numerous capabilities normally not available at the Server System Console, such as copying, renaming, deleting, and purging files. Standard NetWare .NCF files, plus those utilizing TaskMaster's extensions and script language, can be executed either manually via the TMConsole (Shell), the Server Console, or as a scheduled task.

The Secure Server module provides a more secure environment for remote site processing. It implements scheduled task execution and enhanced batch processing, as well as supporting authenticated Remote Server Console access, with the following limitations:

- Most Console Commands extensions (both local and remote) are only supported for batch processing (i.e., they cannot be executed via the TMConsole or the System Console prompt but are supported within a task)

- Batch processing can only be initiated either as a scheduled task or as a remote command submitted by a Master Server Module (i.e., tasks cannot be executed directly via the TMConsole or the System Console).

The Secure Server module is typically used at remote locations or on Servers where physical access may not be easily restricted.

TaskMaster Lite (TMLite)

TaskMaster Lite (TMLite) consists of a single Server module:

- TMLite Server module (TMLITE.NLM)

All the functionality in TMLite is provided by the Server module.

Loading the Server Module

Initial execution of the Server module requires access to the Server's System Console. This can be accomplished either by physically going to the Server or through Novell's RCONSOLE utility.

To load the Server module, type one of the following at the console:

LOAD [vol:path\]TASKMSTR	(full TaskMaster Master NLM)
LOAD [vol:path\]TMSECURE	(full TaskMaster Secure NLM)
LOAD [vol:path\]TMLITE	(TaskMaster Lite NLM)

If the Server module was installed in the SYS:SYSTEM directory or in one of the defined Server System Console search paths, the volume and path are not required. Otherwise, the volume and path should correspond to the Server files destination directory.

The LOAD command line should be added to the AUTOEXEC.NCF file to insure automatic loading of the Server module during subsequent rebooting of the File Server.

Notes: If this is the first time the Server module has been loaded, an informational alert may indicate that the configuration file could not be found. This file is created once the first task is scheduled.

It is not possible to load both the Master and Secure Server modules.

Command Line Options

The following command line options can be used when loading the Server module.

IP=#.#.#.#

Specifies an IP Address (e.g., IP=192.168.1.1) to be used for Server-to-Server communications. If not specified, the primary (or first specified) Server IP Address is used. This option should only be used if required to override the defaults due to IP configuration conflicts or traffic constraints.

P=[vol:path]
PATH=[vol:path]

Specifies an additional path that is to be searched when attempting to open task files for processing. If only a path is specified (i.e., no volume), it is regarded as relative to the root of the SYS: volume.

Notes: If specified, this is the first path searched. If not specified or the task file is not found, the Server module will first search the startup directory (i.e., the directory from which the Server module was loaded) then the Server's system directory (SYS:\SYSTEM).

Not supported by the Secure Server module.

SECURE

Specifying SECURE limits the acceptance of remote requests to only those received from Master Server modules with identical serialization to this Server module.

Note: Not supported by the TaskMaster Lite (TMLite) Server module.

Example:

LOAD TASKMSTR SECURE
Only a Master Server module with identical serialization can submit remote Server requests to this Server module.

Execute Task Upon Load

Specifying a task file on the command line during the Server module load will result in the task being executed once the Server module is completely loaded.

Examples:

LOAD TASKMSTR SYS:SYSTEM\STARTUP.TSK
LOAD TASKMSTR STARTUP.TSK
If no path is provided in the task file specification, the directory from which the Server module was loaded is searched, followed by the Server System directory, and then the optional task file directory (P=/PATH=, if specified).

Note: The Secure Server module only supports this option during the initial five (5) minutes of Server up-time (i.e., since the Server was last restarted) to facilitate its use in AUTOEXEC.NCF. If the Server has been running for more than five (5) minutes, this option is ignored by the Secure Server module.

Name Space Support

Multi-OS support for DOS, Macintosh / AFP, Long (OS2 / Windows), and NFS (Unix) Name Spaces is inherent, unless documented to the contrary with maximum combined full path and name lengths of up to 1023 characters. Support for UTF-8 naming in non-DOS Name Spaces.

For full Name Space support to be available on non-NSS Volumes, the appropriate Name Space support NLM must be loaded on the Server (both source and destination Servers for Server-to-Server operations):

NetWare v6.x/OES:	MAC.NAM	(Macintosh / AFP)
	NFS.NAM	(Unix NFS)

Support for non-DOS Name Spaces is similar to the support for the Long (Win32) Name Space in a DOS box under Windows 95 / 98 / NT / 2000 with the DOS Name Space as the default and spaces within a command assumed to be command line item separators. To specify a non-DOS name containing one or more spaces within a command, the entire non-DOS specification must be enclosed in double quotes ("").

Examples:

```
DIR  "SYS:\Program Files\Internet Explorer\default\*.*"
COPY  "SYS:\Program Files\*.*"  "BACKUP:\Program Files"
COPY  "SYS:\Program Files\*.*"   BACKUP:\PROGA~1
```

Multi-OS support is not available without caveats, conflicts and limitations. The greatest conflicts concern the fact that each Name Space supports different standards (or rules) for constructing file names and directory paths. (As the old saying goes, "*The nice thing about standards is that there are so many to choose from*.")

Long Name Space Issues:

According to the standards (or rules) for Windows Long Name Space support (the most widely used Long Name Space implementation), directory and file names may contain up to 255 characters, including spaces but excluding the following characters: Forward slash (/), backward slash (\), colon (:), greater than sign (>), less than sign (<), vertical bar (|), double quotes (“), asterisk (*), and question mark (?). The colon (:) is reserved for use as the drive or volume (NetWare) separator while backward slashes (\) are reserved for use as the UNC or path separator. The forward slash (/) is also reserved and considered a path separator for compatibility with NetWare and Unix NFS file systems. All of the other reserved characters have special purposes in DOS and Windows command line processing so they are not allowed in directory or file names.

Mac Name Space Issues:
According to the standards (or rules) for the Macintosh File System, the only restricted character in a file or folder (directory) name is the colon (:) which is reserved for use as a path separator. Characters normally used as wild cards, such as the asterisk (*) and question mark (?), can be used within a file or folder (directory) name. Characters which have special meanings on many Operating Systems, such as greater than (>) and less than (<) signs often used for input or output redirection, or double quotes (“) commonly used to enclose a specification with embedded spaces and the ampersand (&) widely used widely in internet applications as an options separator on the command line, are valid with a file and folder (directory) name. In addition, forward slashes (/) which are reserved as a path separator by both NetWare and NFS standards (or rules), as well as backward slashes (\) which are reserved as a path separator by DOS, Long (Windows), and NetWare, are allowed in Macintosh file and folder (directory) names. The only real limitation is that a file or folder (directory) name cannot contain more than 31 characters, but a full path can exceed the 255 character DOS limitation.

NFS Name Space Issues:
According to the standards (or rules) for the NFS File System, the only restricted character in a file name or path is the forward slash (/) which is reserved for use as a folder (path) separator. Characters normally used as wild cards, even for Unix commands, such as the asterisk (*) and question mark (?), can be used within a file or directory name. In addition, colons (:), which are used by DOS, Long (OS/2 / Windows), and NetWare to separate Drives/Volumes from Paths, can be used in file or directory names. NFS also allows backward slashes (\), greater than (>) and less than (<) signs, as well as double quotes (“) and ampersands (&) to be used in file and directory names. Moreover, both file and directory names are handled as case sensitive, treating FILE, File, and file as three separate names. The only real limitation is a maximum of 255 characters per file or directory name allowing a full path to far exceed the 255 character DOS limitation.

Every effort has been made and will continue to be undertaken to insure that TaskMaster / TaskMaster Lite can provide as comprehensive of support as possible for all the special cases of each Name Space without being limited to supporting a single Name Space (i.e., maintaining multi-OS support is the Priority). In that regard, there are likely to be some situations where file or directory names will conflict with either the NetWare APIs used or the multi-OS support logic and the entry will not be capable of being processed by TaskMaster / TaskMaster Lite until it has been renamed.

To avoid potential Name Space caveats and conflicts, it is recommended that multi-OS compatible rules be implemented for file and directory naming. Ultimately, in any realm of conflict, it is often best to avoid such problems by defining rules which embrace the path of least resistance (i.e., the lowest common denominator, in this case the Long Name Space).

The following simple rules should provide a reasonable compromise that also insures directories and files are fully accessible and manageable across the network:

- Avoid using any characters in directory or file names which may have special application in other Operating Systems or Name Spaces supported by the network, such as the asterisk (*), question mark (?), colon (:), forward slash (/), backward slash (\), less than sign (<), greater than sign (>), and ampersand (&) in particular. Being descriptive is good but remember that a good file name is also easy to remember and access so keep it brief and simple. It is also important to note that longer names consume more disk space, require more resources to process, complicate backups, waste network bandwidth, and may be incompatible with some applications.

- Avoid using spaces in directory and file names. While spaces within directory and file names are supported by Long, Mac, and NFS Name Spaces, such habits can create conflicts for parsers which recognize spaces as command line option separators. A more preferable method is to use the underscore (_) or a period (.) to separate words within directory or file names, such as my_file.doc.

  Note: While the Macintosh and Unix Operating Systems allow directory and file names to have leading spaces, not all TaskMaster / TaskMaster Lite commands (especially SxCOPY and SYNC) are able to properly process such entries due to limitations in the NetWare API set. Moreover, such names are not supported by either the DOS or Windows Operating Systems. Therefore, the use of leading spaces in directory and file names is neither fully supported nor recommended.

- One final suggestion, not specific to TaskMaster / TaskMaster Lite: Try to use the proper common or widely used extensions. Without a proper extension, some applications and browsers may not be able to recognize the file.

Users should be advised of the importance of adhering to multi-OS compatible rules both for file system management (i.e., backup, restore, disaster recovery, maintenance, etc.) and for transportability of information between systems as networks become more and more heterogeneous.

TMConsole (Shell)

When the TaskMaster Master Server module (TASKMSTR.NLM) or TaskMaster Lite Server module (TMLITE.NLM) is loaded, a TMConsole (Shell) screen is automatically created on the Server. The TMConsole (Shell) provides an interactive user interface to TaskMaster and the TaskMaster Console Command extensions (refer to the Console Commands chapter later in this manual). The combination of the TMConsole (Shell) interface and TaskMaster's Console Command extensions is intended to emulate a DOS command shell (or DOS box under Windows) providing direct command line management of the directories and files stored on volumes mounted on the local Server. TaskMaster's Console Command extensions also include commands for configuring and managing TaskMaster, as well as performing operations against Remote Servers which are running compatible versions of the TaskMaster NLM.

Notes: The TMConsole (Shell) is not supported by the TaskMaster Secure Server module (TMSECURE.NLM).

On Servers running the TaskMaster Secure Server module, Console Command extensions can only be executed as part of scheduled or submitted tasks, not via the Server Console prompt or NetWare batch files.

Server-to-Server operations are not supported by TaskMaster Lite.

Access

Once the TaskMaster Master Server module (TASKMSTR.NLM) or TaskMaster Lite Server module (TMLITE.NLM) is loaded, the TMConsole (Shell) screen can be accessed by changing screens in the normal manner. Entering TMCONSOLE at the Server System Console prompt will change directly to the TMConsole (Shell) screen.

Disable / Enable

Entering EXIT at the TMConsole (Shell) prompt will close the active screen. To re-activate an inactive TMConsole (Shell) prompt, enter TMCONSOLE at the Server System Console.

Send Commands

TaskMaster Console Commands can be sent to the TMConsole (Shell) prompt from the Server's System Console or a NetWare processed .NCF batch file via the TMCONSOLE command as follows:

TMCONSOLE [command]

Example:

TMCONSOLE TMRUN SUBTASK.TSK
When executed either from the Server's System Console or within a NetWare .NCF batch file, the TMRUN SUBTASK.TSK command will be executed as if manually entered at the TMConsole (Shell) prompt.

Specific Commands

The following commands are specific to the TMConsole (Shell) prompt and perform as indicated:

CLS	- Clear the TMConsole (Shell) screen
EXIT	- Close the TMConsole (Shell) screen
SYSTEM CONSOLE	- Change directly to the Server's System Console screen

Task (.TSK)

A task (or script file) is merely an ASCII text file containing processing instructions which are either natively supported by the Server or can be interpreted by the Server module processing the file. The batch processing routine in the Server module reads the task file, spawning a separate thread and environment for use by each task, and then processes the task file contents (or script language) line by line.

The special commands and logic, as well as the Dynamic Variables and Environment Variables, supported by the Server module batch processor are automatically interpreted and handled. In addition, the batch processor recognizes and handles any Batch and TMConsole Command extensions supported by the Server module. Any other lines which could not be interpreted by the batch processor are passed directly to the Server's System Console.

There is no special file format or header required for a task file. Task files can be created and edited using any ASCII text editor. A task file can even create or write another task file and then execute it.

Tasks can be scheduled for automatic execution, launched manually, or launched (called) by another task. With the full TaskMaster, tasks can also be submitted to another Server (via task or remote Client).

Default Search Path

The following search path criteria will be used when a Server module attempts to process a task:

- Search the default directory from which the Server module was loaded for:
  task; task with the .NCF extension (NetWare Server default); task with the .TSK extension.

- Search the SYS:SYSTEM directory for:
  task; task with the .NCF extension (NetWare Server default); task with the .TSK extension.

- If the Master Server module is loaded and an additional search path has been provided when it was loaded (P=/PATH=), search the specified directory for:
  task; task with the .NCF extension (NetWare Server default); task with the .TSK extension.

Notes: For security purposes, the task file associated with a scheduled task must reside within the Default Search Path Criteria. The same restrictions apply for tasks launched via the full TaskMaster Clients. Only tasks initiated via the TMConsole, the System Console, called by a task, or submitted to a Remote Server via the TMSCMD command can include a path within the task file specification.

A task file must reside on the Server for the Server module to be able to process it (i.e., task files on Client workstations are inaccessible)

Hint: 'Cannot locate task file' errors usually occur because the task file does not reside within the Default Search Path Criteria or the task file specification is invalid.

Schedule a Local Task

The Server module sets up a low priority thread which checks at the start of each minute for any tasks scheduled for execution at that time. Multiple tasks can be scheduled for execution at the same time and, in such cases, will execute concurrently. The minimum granularity for task scheduling, by scheduled time or interval, is one minute.

Tasks can be scheduled for execution on a specific date, day of the month, one or more days of the week, at a particular time, at a certain minute after each hour, and at intervals (every xx minutes). Both standard Server System Console batch files (.NCF) and extended TaskMaster tasks (.TSK) can be scheduled.

Note: Task file must adhere to Default Search Path Criteria.

Three types of task schedules are supported:

Date Task	Performed on a specific date (month/day)
Weekly Task	Performed one or more days each week (Sunday through Saturday)
Monthly Task	Performed on a specific day of the month (01 through 31)

Each task schedule type has the added flexibility to utilize one of three supported time formats:

Specific	At a given time (12:00pm)
Every ??	At intervals of ?? minutes
?? After Hr	At ?? minutes after each hour

Tasks can be scheduled via the Task Management Windows Client (TMClient) or the Console Command - Local Server Commands TMSCHEDULE command. Refer to the appropriate section of the manual for information on each option.

Execute a Local Task

A task can be executed from the TMConsole (Shell) prompt (TaskMaster Master or TaskMaster Lite Server modules only) or by another task using the following syntax:

TMRUN [[vol:]path\]{task} [arg] [arg] [arg] ...

Notes: If the task specification includes a path (vol:path):

- Search only the specified directory for:
  task; task with the .NCF extension (NetWare Server default); task with the .TSK extension.

If the task specification does not include a path:

- task must meet Default Search Path Criteria.

Up to ten (10) arguments or parameters ([arg]) can be passed to the task via the command line. Any such [arg] specified on the command line will be placed in the reserved Dynamic Variables (%0 - %9) in the order specified on the command line.

If a task uses TMRUN to execute another Task, both tasks process concurrently.

This command is not supported if the Secure Server module is loaded.

Refer to the Console Commands - Local Server Commands section of the manual for more information on the TMRUN command.

Hint: If an argument [arg] to be passed to a task has embedded spaces, enclose [arg] within double quotes. Otherwise, any spaces encountered may be considered [arg] separators with indeterminate results.

Example:

TMRUN {task} arg0 Arg1 “Argument 2"
The Server module uses the default Search Path criteria to locate the task file (either task, task.NCF, or task.TSK). If located, the task will be read and processed with the following reserved Dynamic Variables initialized as indicated:

%0 = arg0, %1 = Arg1, %2 = Argument 2

Call a Local Task

A task can CALL another task using the following syntax:

CALL [[vol:]path\]task [arg] [arg] [arg] ...

Note: Identical to TMRUN, except that CALL only works from within a task. The task issuing the CALL is suspended until the called task completes. Refer to the Batch Processing - Commands section of the manual for more information on the CALL command.

Submit a Remote Task

If a full TaskMaster Server module is loaded, a task can be submitted to a Remote Server via the TMConsole or the System Console prompt (Master Server module only) or by another task (either Server module) using the following syntax:

TMSCMD RServer TMRUN [[vol:]path\]{task} [args]

Notes: TMSCMD is a TaskMaster Server Console command extension which can be used to submit commands (TMRUN {task} [arg] ...) to a Remote Server (RServer).

Task file specification and Default Search Path Criteria are relative to the Remote Server to which the task was submitted (i.e., the specified task file must already reside on the Remote Server - RServer).

Tasks can also be submitted via the Clients (refer to the appropriate Client chapter for more information).

Remote Server must have a TaskMaster Server module loaded.

Not supported by the TMConsole or the System Console if the Secure Server module is loaded.

Not supported by TaskMaster Lite (TMLite).

Refer to the Console Commands - Remote Server Command section of the manual for more information on the TMSCMD command.

Client Launch of a Task

If a full TaskMaster Server module is loaded, a task can be launched via the TaskMaster Remote Console DOS Client by users with sufficient rights (Supervisor or Admin equivalent, TaskMaster Operator, or TaskMaster User). Refer to the Remote Console DOS Client chapter for more information on launching tasks in this manner.

Task Management Windows Client

TaskMaster's Task Management Windows Client (TMClient) provides the means to monitor, schedule and terminate Tasks via a workstation. Both NetWare .NCF and extended TaskMaster .TSK Tasks can be scheduled for execution on a specific date, day of the month, one or more days of the week, at a particular time, at a certain minute after each hour, and at intervals (every xx minutes), as well as monitored and terminated.

Note: A TaskMaster Server Module must be loaded on the Server for task management support.

Remote Console DOS Client

TaskMaster's Remote Console DOS Client (TMRemote) provides secure and controlled Server Console access. Although it is a DOS program, it can be run in multiple DOS boxes under Windows allowing for multiple concurrent secure Remote Console sessions with a single or multiple Servers. Through configuration options, it is possible to provide Remote Console access to specific users in the range of full to limited (view only) rights. It also supports command line options which support launching tasks and submit commands to Servers from a workstation. This combination of features make it a versatile compliment to the Server Module.

Note: A TaskMaster Server Module must be loaded on the Server for remote console support (refer to the chapter titled Remote Console DOS Client for more information).

To define/review the Remote Console access rights and access password for TMRemote sessions, use

TaskMaster's TMCONFIG extended Console Command.

The Remote Console DOS Client (TMRemote) is not supported by TaskMaster Lite (TMLite).

TaskMaster Operators

By default, (NDS) Admin or (Bindery) Supervisor, or equivalent, rights are required to schedule tasks or use the TaskMaster Remote Console (TMRemote) DOS Client (TMREMOTE.EXE) to establish a remote console session, launch tasks from a workstation, and submit commands to a Server. However, it is possible to define TaskMaster Operators who are provided with scheduling privileges and can be granted restricted access to the Server Console via TMRemote.

To define/review TaskMaster Operators for a Server, use the TMCONFIG extended Console Command.

Note: TaskMaster Operator definitions are specific to each Server and are not exchanged between Servers.

TaskMaster Operators can only be defined by an (NDS) Admin or (Bindery) Supervisor, or equivalent user.

Not supported by TaskMaster Lite (TMLite).

TaskMaster Users

By default, (NDS) Admin or (Bindery) Supervisor, or equivalent, rights are required to use the TaskMaster Remote Console (TMRemote) DOS Client (TMREMOTE.EXE) to establish a remote console session, launch tasks from a workstation, and submit commands to a Server. However, it is possible to define TaskMaster Users and grant them restricted access when using TMRemote.

To define/review TaskMaster Users for a Server, use the TMCONFIG extended Console Command.

Note: TaskMaster User definitions are specific to each Server and are not exchanged between Servers.

TaskMaster Users can only be defined by an (NDS) Admin or (Bindery) Supervisor, or equivalent user.

Not supported by TaskMaster Lite (TMLite).

[INTENTIONAL BLANK PAGE]

Chapter 3 - Batch Processing

The **TaskMaster**® script language combines the simplicity found in of DOS batch and NetWare System Login scripts with the power of a programming language. Tasks scripts can consist of standard NetWare Server System Console Commands, TaskMaster's TMConsole (Shell) extended Console Commands, TaskMaster's extended Batch Commands, or a combination of all three, plus TaskMaster's enhanced scripting logic.

The enhanced scripting logic incorporates IF/ELSEIF/ENDIF, WHILE/LOOP, and CALL/GOTO execution path constructs supporting AND/OR conditional testing and string comparisons (<=, <, ==, >, >=). The integration of the extended Batch and Console Commands, conditional tests, Dynamic Variables and Environment Variables provide the functionality necessary to pro-actively monitor, manage, and manipulate Server resources. Through this combination of logic and information resources, complicated and sophisticated tasks become simple, automated operations.

With respect to the documentation in this chapter, the following points should be noted:

- All structures and commands are case insensitive, unless otherwise noted.
- Indentations in the examples are provided for readability and are not required.
- Items separated by a vertical bar (|) indicate any one of the items may be used, but not more than one (either or).
- Items in brackets [ ] are optional. (Brackets not required.)
- Items in curly braces { } are required. (Braces not required.)
- Lines beginning with REM (case insensitive) or a non-alphanumeric character are treated as comments and are ignored during processing (see next note on labels).
- Lines beginning with a colon : are considered labels for GOTO execution.

Environment Variables

The enhanced Batch Processor supports the following Environment Variables that will be automatically substituted prior to the script line being processed:

%AM_PM%
%CONN_ADDRESS_#%
%CONN_ID_#%
%CONN_NAME_#%
%CONNS_ACTIVE%
%CONNS_IN_USE%
%CONNS_MAX%
%CONNS_PEAK%
%CONSOLE_SCREEN%
%CPU_UTIL%
%CURRENT_SCREEN%
%CWD%
%CX%
%DAY%
%DAY_OF_WEEK%
%DIR_FILE_%
%DIR_SUB_%
%DIR_TREE_%
%DOS_DRIVE_FREE_%
%DOS_DRIVE_SIZE_%
%DOS_DRIVE_USED_%
%ELAPSED_HOURS%
%ELAPSED_MINS%
%ELAPSED_SECS%
%ELAPSED_TIME%
%FILE_ACCESS_?%
%FILE_ATTRIB_?%
%FILE_CREATE_?%
%FILE_MODIFIER_?%
%FILE_NAME_DOS_?%
%FILE_NAME_LONG_?%
%FILE_NAME_MAC_?%
%FILE_NAME_NFS_?%
%FILE_OWNER_?%
%FILE_SIZE_?%
%FILE_UPDATE_?%
%HOUR%
%HOUR24%
%MINUTE%
%MONTH%
%MONTH_NAME%
%NDAY_OF_WEEK%
%NDAY_OF_YEAR%
%NW_SUBVERSION%
%NW_SUPPORTPACK%
%NW_VERSION%
%PATH%
%SCREEN_NAME%
%SECOND%
%SERVER%
%SERVER_IP%
%SERVER_NET%
%TASK%
%TASK_EXT%
%TASK_FILE%
%TASK_LINE_NUM%
%TASK_NAME%
%TASK_PATH%
%TASK_SCREEN%
%TEMP_NAME%
%TM_REVISION%
%TM_SUBVERSION%
%TM_VERSION%
%TREE%
%UPTIME_NUMERIC%
%UPTIME_STRING%
%VOL_FREE%
%VOL_NAME%
%VOL_PURGEABLE%
%VOL_SIZE%
%VOL_SIZE_MB%
%VOL_USED%
%YEAR%

Examples:

ECHO %MONTH%/%DAY%/%YEAR% %HOUR%:%MINUTE%
Echoes the current date and time (e.g., 12/31/2002 06:00).

ECHO TaskMaster v%TM_VERSION%.%TM_SUBVERSION%
Echoes the TaskMaster release (e.g., TaskMaster v3.19)

IF %HOUR24%>07 AND %HOUR24%<18 THEN ECHO WORK
Echoes the word WORK between the hours of 8am and 5pm.

%AM_PM%

Substitutes ‘am’ or ‘pm’, based upon the Server’s 12 hour local time.

%CONN_ADDRESS_#%

Substitutes the IP/IPX address for the specified connection number (#). The specification (#) must be a valid connection number. If the connection is INACTIVE, a zero filled address is returned.

Note: If a Dynamic Variable is used for ‘#’, it must have previously been defined as a valid connection number.

Examples:

```
ECHO Conn: 1  %CONN_NAME_1%  [%CONN_ID_1%]  @  %CONN_ADDRESS_1%
```

Display (ECHO) the logged in name of the User (%CONN_NAME_1%), Object ID (%CONN_ID_1%), and the Network Address (%CONN_ADDRESS_1%) for connection #1.

```
DEFINE %0 01

WHILE “%0"<“10"
    ECHO Conn: %0  %CONN_NAME_%0%  [%CONN_ID_%0%]  @  %CONN_ADDRESS_%0%
    DEFINE %0 %0+=01
LOOP
```

Dynamic Variable %0 is defined as 01, a starting point for a list of connection addresses. The condition WHILE/LOOP structure will process the interior lines as long as %0 is less than ‘10' (hex string comparison of the variables). The Connection number (%0), logged in name of the User (%CONN_NAME_%0%), Object ID (%CONN_ID_%0%), and the Network Address (%CONN_ADDRESS_%0%) are displayed via the ECHO command. %0 is then incremented by 1 (DEFINE %0 %0+=01) for the next connection.

%CONN_ID_#%

Substitutes an ASCII string containing the Object ID of the currently logged in User for the specified connection number (#). The specification (#) must be a valid connection number. If the connection is NOT-LOGGED-IN or INACTIVE, the value of 0 is returned.

Notes: The substituted string is the same hexadecimal value used for the SYS:\MAIL directory for the logged in User.

If a Dynamic Variable is used for ‘#’, it must have previously been defined as a valid connection number.

Refer to %CONN_ADDRESS_% for examples.

%CONN_NAME_#%

Substitutes the name of the logged in User for the specified connection number (#). The specification (#) must be a valid connection number. If the User has logged out of the connection but the slot has not yet been released, NOT-LOGGED-IN is returned. If the connection slot is free (not allocated), INACTIVE is returned.

Notes: If a Dynamic Variable is used for ‘#’, it must have previously been defined as a valid connection number.

Refer to %CONN_ADDRESS_% for examples.

%CONNS_ACTIVE%

Substitutes the current active (allocated) connection count (numeric value of 0000 to 9999 - 4 digit fixed length).

Note: Includes Licensed, Authenticated, NOT-LOGGED-IN, NLM, and remote attachment connections.

%CONNS_IN_USE%

Substitutes the current active (allocated) connections in use count (numeric value of 0000 to 9999 - 4 digit fixed length).

Note: Note: Includes Licensed, Authenticated, NLM, and remote attachment (excludes NOT-LOGGED-IN) connections.

%CONNS_MAX%

Substitutes the maximum number of active connections since the NLM was loaded (numeric value of 0000 to 9999 - 4 digit fixed length).

%CONNS_PEAK%

Substitutes the maximum number of active connections since the NLM was loaded (numeric value of 0000 to 9999 - 4 digit fixed length).

%CONSOLE_SCREEN%

Substitutes the name of the current screen actively displayed on the Server Console.

%CPU_UTIL%

Substitutes current Server CPU utilization percentage (numeric value of 000 to 100 - 3 digit fixed length).

%CURRENT_SCREEN%

Substitutes the name of the current screen against which the active task performs keyin / screen interrogation commands.

%CWD%

Substitutes Current Working Directory (CWD).

Note: The CWD is the default vol:path (directory) for the active task (use the CD/CHDIR command to alter the CWD).

Examples:

SYS:\PUBLIC\IBM_PC\MSDOS\V6.23
SYS: (root dir of SYS: Volume, i.e., SYS:\)

%CX%

Substitutes the Directory Services Context for the Server.

%DAY%

Substitutes current day of the month (01 to 31 - 2 digit fixed length).

%DAY_OF_WEEK%

Substitutes current day of the week (Sunday - Saturday).

%DIR_FILE_{spec}%

Substitutes the DOS name of the first/next file which matches {spec}.

Notes: {spec} must be a valid DOS search pattern (path specific or relative to the Current Working Directory - CWD), for the initial (first) request and null for subsequent (next) requests (i.e., specifying a pattern initializes the search). A null string is returned if {spec} does not exist or the search is complete.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid file search pattern.

Wild cards supported.

Examples:

```
DEFINE %0 %DIR_FILE_SYS:SYSTEM\*.*%
```

Defines Dynamic Variable %0 as the DOS entry name for the first matching file in the CWD.

```
DEFINE %0 %DIR_FILE_%
```

Defines Dynamic Variable %0 as the DOS entry name for the next matching file in the CWD (requires an initial successful match of the {spec} pattern).

```
// List files in a specified directory

DEFINE %1 %DIR_FILE_SYS:SYSTEM\*.*%

WHILE “%1">””
    ECHO %1
    DEFINE %1 %DIR_FILE_%
LOOP
```

Dynamic Variable %1 is defined as the result of the first (initial) search for files matching {spec} (SYS:SYSTEM*.*). The conditional WHILE / LOOP structure compares %1 against a NULL value (“%1">””) before processing the commands within the structure. If %1 is not NULL, the commands within the WHILE / LOOP structure display the file name (ECHO %1) then attempt to retrieve the next (subsequent) file matching {spec} in the directory. The LOOP then returns processing control to the first command after the matching WHILE. When no more matching files exist, %1 is returned as a NULL value which terminates the conditional WHILE / LOOP.

```
// List dirs and files in a subdirectory tree

DEFINE %0 SYS:SYSTEM
CD %0

WHILE
    ECHO %CWD%

    DEFINE %1 %DIR_SUB_*.*%
    WHILE “%1">”“
        ECHO [%1]
        DEFINE %1 %DIR_SUB_%
    LOOP

    DEFINE %1 %DIR_FILE_*.*%
    WHILE “%1">”“
        ECHO %1
        DEFINE %1 %DIR_FILE_%
    LOOP

    DEFINE %1 %DIR_TREE_%0%
    DEFINE %0

    IF “%1"==”“ THEN BREAK
LOOP
```

Dynamic Variable %0 is defined as the starting directory and the Current Working Directory (CWD) is changed to the same location (CD %0). The commands in the outermost WHILE / LOOP display the CWD (ECHO %CWD%) then defines %1 as the result of the first (initial) subdirectory search (%DIR_SUB_*.*% since a VOL:PATH was not included in the {spec} the search defaults to the CWD). A nested conditional WHILE / LOOP compares the result in %1 against a NULL value (“%1">”“) before processing the commands within the structure. If %1 is not NULL, the commands within the WHILE / LOOP structure display the subdirectory name (ECHO [%1]) then attempt to retrieve the next (subsequent) directory in the CWD which matches {spec - *.*}. When no more matching directories exist, %1 is returned as a NULL value which terminates the conditional WHILE / LOOP. Control returns to the outermost WHILE / LOOP which defines %1 as the result of the first (initial) file search (%DIR_FILE_*.*% since a VOL:PATH was not included in the {spec} the search defaults to the CWD). A nested conditional WHILE / LOOP compares %1 against a NULL value (“%1">”“) before processing the commands within the structure. If %1 is not NULL, the commands within the WHILE / LOOP structure display the file name (ECHO %1) then attempt to retrieve the next (subsequent) file in the CWD which matches {spec - *.*}. When no more matching files exist, %1 is returned as a NULL value which terminates the conditional WHILE / LOOP. Control returns to the outermost WHILE / LOOP which defines %1 as the result of the tree search (%DIR_TREE_%0%), initially (first) using the starting location in %0 which is then cleared (DEFINE %0) for subsequent (next) searches. If the tree search returns a NULL value, indicating there are no more levels of the subdirectory tree to traverse, the conditional test (“%1"==”“) becomes TRUE and terminates the WHILE / LOOP (BREAK). Otherwise, the returned subdirectory becomes the Current Working Directory (CWD) and LOOP then returns processing control to the first command after the matching WHILE.

%DIR_SUB_{spec}%

Substitutes the DOS 8.3 name of the first/next directory in the which matches {spec}.

Notes: {spec} must be a valid DOS search pattern (path specific or relative to the Current Working Directory - CWD), for the initial (first) request and null for subsequent (next) requests (i.e., specifying a pattern initializes the search). A null string is returned if {spec} does not exist or the search is complete.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid dir search pattern.

Wild cards supported.

Refer to %DIR_FILE_% for examples.

%DIR_TREE_{spec}%

Substitutes the DOS 8.3 name of the first/next directory in the location designated by {spec} and changes the Current Working Directory (CWD) to the directory. Subsequent (next) requests will traverse the directory tree using {spec} as the base (or root) and returning all the directory entries in the directory tree below {spec}.

Notes: {spec} must be a valid DOS search pattern (path specific or relative to the Current Working Directory - CWD), for the initial (first) request and null for subsequent (next) requests (i.e., specifying a pattern initializes the search). A null string is returned if {spec} does not exist or the search is complete.

The initial (first) specification becomes the base (or root) directory for traversing the tree.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory.

Unlike %DIR_SUB_% which only substitutes the directory name, the directory also becomes the CWD.

Refer to %DIR_FILE_% for examples.

%DOS_DRIVE_FREE_?%

Substitutes the amount of free space (MBytes) on the specified or default (boot) DOS drive (0000 to 99999999 - 4 to 8 digits variable length). If a DOS drive is not specified (i.e., ? is null instead of a drive letter such as A-Z), the default (boot) DOS drive information is returned.

%DOS_DRIVE_SIZE_?%

Substitutes the drive size (MBytes) for the specified or default (boot) DOS drive (0000 to 99999999 - 4 to 8 digits variable length). If a DOS drive is not specified (i.e., ? is null instead of a drive letter such as A-Z), the default (boot) DOS drive information is returned.

%DOS_DRIVE_USED_?%

Substitutes the amount of used space (MBytes) on the specified or default (boot) DOS drive (0000 to 99999999 - 4 to 8 digits variable length). If a DOS drive is not specified (i.e., ? is null instead of a drive letter such as A-Z), the default (boot) DOS drive information is returned.

%ELAPSED_HOURS%

Substitutes the number of hours that the current task has been running (00 to 4294967295 - 2 digits minimum length).

%ELAPSED_MINS%

Substitutes the number of minutes that the current task has been running (00 to 4294967295 - 2 digits minimum length).

%ELAPSED_SECS%

Substitutes the number of seconds that the current task has been running (00 to 4294967295 - 2 digits minimum length).

%ELAPSED_TIME%

Substitutes a text string representing the hours, minutes, and seconds that the current task has been running (00:00:00 to 99999:59:59 - minimum length of 8 bytes).

%FILE_ACCESS_{spec}%

Substitutes the date {spec} was last accessed (yyyy/mm/dd).

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Examples:

ECHO SYS:\SYSTEM\SYS$LOG.ERR was last accessed: %FILE_ACCESS_SYS:\SYSTEM\SYS$LOG.ERR%

Displays (ECHO) the last accessed date for the specified file (SYS:\SYSTEM\SYS$LOG.ERR).

DEFINE %0 SYS:\SYSTEM\SYS$LOG.ERR

ECHO DOS:	%FILE_NAME_DOS_%0%	Owner:	%FILE_OWNER_%0%
ECHO Long:	%FILE_NAME_LONG_%0%	Attributes:	%FILE_ATTRIB_%0%
ECHO Mac:	%FILE_NAME_MAC_%0%		
ECHO NFS:	%FILE_NAME_NFS_%0%		
ECHO Created:	%FILE_CREATE_%0%	Size:	%FILE_SIZE_%0%
ECHO Updated:	%FILE_UPDATE_%0%	By:	%FILE_MODIFIER_%0%
ECHO Access:	%FILE_ACCESS_%0%		

Displays the %FILE_...% available information for the file defined in Dynamic Variable %0 (DEFINE).

%FILE_ATTRIB_{spec}%

Substitutes the attributes {spec} has been assigned.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

For information on the returned attributes and their order, refer to the CHMOD / FLAG Extended Console Commands.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_CREATE_{spec}%

Substitutes the date {spec} was created (yyyy/mm/dd).

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_MODIFIER_{spec}%

Substitutes the name of the last User to update or modify {spec}.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_NAME_DOS_{spec}%

Substitutes the full DOS Name Space name (vol:\path\filename.ext) for {spec}.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_NAME_LONG_{spec}%

Substitutes the full LONG Name Space name (vol:\path\filename.ext) for {spec}.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_NAME_MAC_{spec}%

Substitutes the full Macintosh (MAC) Name Space name (vol:path:filename.ext) for {spec}.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_NAME_NFS_{spec}%

Substitutes the full NFS Name Space name (vol:/path/filename.ext) for {spec}.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_OWNER_{spec}%

Substitutes the Owner name assigned to {spec}.

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_SIZE_{spec}%

Substitutes the size (numeric value of 0000000000 to 4294967295 - 10 digits fixed).

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%FILE_UPDATE_{spec}%

Substitutes the date {spec} was last updated (yyyy/mm/dd).

Notes: {spec} can be path specific or relative to the Current Working Directory (CWD) and must reference a valid directory or file. If {spec} does not exist, a null string is returned.

{spec} may be a Dynamic Variable provided it has been previously defined as a valid directory or file.

Refer to %FILE_ACCESS_{spec}% for examples.

%HOUR%

Substitutes the current 12 hour format (01 to 12 - 2 digit fixed length).

%HOUR24%

Substitutes the current 24 hour format (00 to 23 - 2 digit fixed length).

%MINUTE%

Substitutes the current minute (00 to 59 - 2 digit fixed length).

%MONTH%

Substitutes the current month number (01 to 12 - 2 digit fixed length).

%MONTH_NAME%

Substitutes the current month name (January to December).

%NDAY_OF_WEEK%

Substitutes a number corresponding to the current week day (Sunday = 1 to Saturday = 7 - 1 digit fixed length).

%NDAY_OF_YEAR%

Substitutes the current Julian day number in the year (001 to 365 [366 in Leap Years] - 3 digit fixed length).

%NW_SUBVERSION%

Substitutes the minor version of the NetWare OS in use (00 to 99 - 2 digit fixed length).

%NW_SUPPORTPACK%

Substitutes the NetWare Support Pack level (returns SP#, # is level).

Note: NetWare v5+ only (NetWare v4 returns a null string).

%NW_VERSION%

Substitutes the major version of the NetWare OS in use (0 to 9 - 1 digit fixed length).

%PATH%

Substitutes the path, excluding the volume name, of the Current Working Directory (CWD).

Notes: SYS:\ returns: (<null> - no path or '\' separator)
SYS:\SYSTEM returns: \SYSTEM

%SCREEN_NAME% Note: Obsolete (supported for backwards compatibility). Replacement: %CONSOLE_SCREEN%

Substitutes the name of the current screen against which the active task performs keyin / screen interrogation commands.

%SECOND%

Substitutes the current second (00 to 59 - 2 digits fixed length).

%SERVER%

Substitutes the name of the Server.

%SERVER_IP%

Substitutes the Server's primary IP network address, if defined (ASCII numeric, full IP format - ###.###.###.###).

%SERVER_NET%

Substitutes the Server IPX Internal Network address, if defined (hex value in ASCII format - 8 byte fixed length).

%TASK%

Substitutes the file name and extension (file.ext) for the active Task.

%TASK_EXT%

Substitutes the extension (no path or file name) for the active Task.

%TASK_FILE%

Substitutes the file name (no path or extension) for the active Task.

%TASK_LINE_NUM%

Substitutes the line number where located within the active Task.

%TASK_NAME%

Substitutes the full name (vol:\path\filename.ext) for the active Task.

%TASK_PATH%

Substitutes the path (vol:\path) for the active Task.

%TASK_SCREEN%

Substitutes the name of the screen created to receive the default I/O, as well as error messages, for the active task.

%TEMP_NAME%

Substitutes a unique temporary NetWare file name (in the format of _TM?????.TMP, where the ?'s are replaced with an ASCII representation of a unique decimal number; e.g., _TM01234.TMP).

%TM_REVISION%

Substitutes the revision level of the TaskMaster module in use (a to z or null - 1 byte max length).

%TM_SUBVERSION%

Substitutes the minor version of the TaskMaster module in use (01 to 99 - 2 digits fixed length).

%TM_VERSION%

Substitutes the major version of the TaskMaster module in use.

%TREE%

Substitutes the Directory Services Tree name in which the Server is located.

%UPTIME_NUMERIC%

Substitutes the current Server Up Time as a numeric string representing the days, hours, minutes, and seconds (000:00:00:00 to 999:23:59:59 - fixed length, zero filled).

%UPTIME_STRING%

Substitutes the current Server Up Time as a text string representing the days, hours, minutes, and seconds ('1 secs' to '999 days 23 hrs 59 mins 59 secs' - variable length).

%VOL_FREE%

Substitutes the percentage of free volume space on the volume associated with the Current Working Directory (CWD), including purgeable space (000 to 100 - 3 digits fixed length).

%VOL_NAME%

Substitutes the name of the volume (VOL:) associated with the Current Working Directory (CWD).

%VOL_PURGEABLE%

Substitutes the percentage of purgeable space on the volume associated with the Current Working Directory (CWD) (000 to 100 - 3 digits fixed length).

%VOL_SIZE%
%VOL_SIZE_MB%

Substitutes the size (MBytes) of the volume associated with the Current Working Directory (CWD) (00000000 to 99999999 - 8 digits fixed length).

%VOL_USED%

Substitutes the percentage of used space on the volume associated with the Current Working Directory (CWD) (000 to 100 - 3 digits fixed length).

%YEAR%

Substitutes the current year (1900 to 2100 - 4 digits fixed length).

Dynamic Variables

The enhanced Batch Processor supports the following Dynamic Variables that will be automatically substituted prior to the script line being processed:

%0 - %9
%VAR00% - %VAR99%

The Batch Processor supports an array of 100 Dynamic Variables (named %VAR00% through %VAR99%, by default) which can be used to store alphanumeric data, as well as modified and queried, during processing. The first 10 of these 100 Dynamic Variables can also be referenced using the following reserved names (%0 through %9) and are used to transfer command line arguments into temporary storage.

Notes: The reserved Dynamic Variable named %0 and the default Dynamic Variable %VAR00% reference the first dynamic variable in the array. Likewise, %9 and %VAR09% reference the tenth dynamic variable in the array.

Upon initial task execution, %0 through %9 (%VAR00% through %VAR09%) are assigned any arguments (options) which are specified on the TMRUN command line. Otherwise, they are set to NULL, as are the remaining Dynamic Variables (%VAR10%-%VAR99%).

Using the Batch Command VARALIAS, it is possible to change the default name (%VAR00% through %VAR99%) used to reference a Dynamic Variable to something which better describes its usage. It is important to note that changing the default name for a Dynamic Variable does not change its array location or data contents, only the default name used to reference the information assigned to the Dynamic Variable. It should also be noted that changing the default name for any of the first 10 Dynamic Variables (%VAR00% through %VAR09%) does not have any affect on the reserved names %0 through %9 assigned to the same Dynamic Variables.

Using the Batch Command DEFINE, it is possible to modify the contents stored within a Dynamic Variable. The contents of the Dynamic Variables can also be queried and tested via several Commands supported by the TaskMaster / TaskMaster Lite (TMLite) enhanced batch processor.

The maximum amount of the data which can be stored within any single dynamic variable is 511 bytes.

Examples:

TMRUN TEST.TSK arg0 Arg1 Argument2 “ARGUMENT 3"
Upon execution:%0 = arg0, %1 = Arg1, %2 = Argument2, %3 = ARGUMENT 3
(%VAR00% = arg0, %VAR01% = Arg1, %VAR02% = Argument2, %VAR03% = ARGUMENT 3)
%4 to %9 (%VAR04% to %VAR09%, plus %VAR10% to %VAR99%), are all defined as NULL.

DEFINE %0 Zero
%0 has been defined as the string ‘Zero’. %VAR00% represents the same Dynamic Variable as %0 so %VAR00% also references the string value ‘Zero’.

VARALIAS %VAR00% %ZERO%
The Dynamic Variable formerly referenced by %VAR00% has been renamed %ZERO%. Because %VAR00% also

referenced the Dynamic Variable %0, %ZERO% now references the same Dynamic Variable as %0 with both defined as the string value ‘Zero’. While %VAR00% no longer exists because it was replaced by %ZERO%, %0 is not renamed because it is a reserved Dynamic Variable name (%0-%9).

DEFINE %ZERO% 0000
%ZERO%, as well as %0 since %VAR00% was renamed to %ZERO%, has been redefined as the string value ‘0000’.

DEFINE %VAR99% 99
VARALIAS %VAR99% %NINETY-NINE%
%NINETY-NINE% has replaced %VAR99% and assumes its data contents (the string value ‘99').

Conditional Structures

The following Conditional Structures can be used within a Task file that will be processed by the TaskMaster / TaskMaster Lite (TMLite) enhanced Batch Processor. Conditional Structures direct processing based upon the result of one or more supported Conditional Tests.

Multiple tests can be evaluated within a single conditional structure evaluation based upon boolean AND / OR operators. The boolean AND operator requires that the test preceding and following must both yield a true result for the evaluation to generate a true condition. The boolean OR operator only requires that either the test preceding or following must yield a true result for the evaluation to generate a true condition.

Notes: Multiple AND / OR operators may be used, providing all such operators are of the same type (AND / OR operators cannot be mixed).

Supported conditional tests are documented separately.

Preceding a test with a NOT operator will generate a true condition if the test fails (i.e., double negative = positive; NOT A==B yields a true condition). Each NOT operator applies only to the specific test which it precedes, not the entire conditional test evaluation.

IF / ELSEIF / ELSE / END

IF / ELSEIF / ELSE / END structures direct command execution based upon the evaluation of one or more test conditions {cond}.

```
IF [NOT] {cond} [GOTO {label}]

IF [NOT] {cond} [THEN command...]

IF [NOT] {cond} [THEN BEGIN]
    command(s)...
ENDIF

IF [NOT] {cond}
    command(s)...
ENDIF

IF [NOT] {cond} [[AND|OR] [NOT] {cond}]
    command(s)...
[ELSEIF [NOT] {cond}
    command(s)...]
[ELSE
    command(s)...]
ENDIF
```

Notes: Supports nested IF / WHILE structures, multiple ELSEIF, & multiple AND|OR logic (cannot mix AND/OR logic).

Dynamic Variables and/or Environment Variables can be used within {cond} test variables.

IF / ELSEIF conditions are evaluated sequentially (in order).

Examples:

```
// Check if the current task is already running
IF ACTIVE_TASK %TASK% THEN EXIT

// Check if weekend (%NDAY_OF_WEEK%, Sun=1 - Sat=7)
IF %NDAY_OF_WEEK%==1 OR %NDAY_OF_WEEK%==7
    // Check if file is in use (FILE_IN_USE)
    IF FILE_IN_USE APP:\DB\CUST.DB
        // Close it (no one should use it on weekend).
        FILE CLOSE APP:\DB\CUST.DB
    ENDIF
ENDIF

// Check if New Years Day (01/01)
IF %MONTH%==01 AND %DAY%==01 THEN ECHO New Year

// Hourly motivational message to Users (nested IF structs)
IF %MINUTE%==00
    IF NOT %HOUR24%>07 OR NOT %HOUR24%<18
        // Not 8:00am - 5:00pm so skip.
    ELSEIF %HOUR24%==12
        SEND "*Time to Eat*"
    ELSEIF %HOUR24%==17
        SEND "** Go Home **"
    ELSE
        SEND "Back to Work!"
    ENDIF
ENDIF
```

WHILE / BREAK / CONTINUE / LOOP

WHILE / LOOP structures provide repetitive execution of commands based upon evaluation of one or more test conditions {cond}. Commands within the WHILE / LOOP structure will continue to be executed, with the process looping back to the top of the WHILE structure, as long as the evaluation remains true. If the evaluation fails, execution proceeds to the first line following the LOOP.

```
WHILE
    command(s)...
    [IF [NOT] {cond} THEN] BREAK
    [IF [NOT] {cond} THEN] CONTINUE
LOOP

WHILE [NOT] {cond}
    command(s)...
    [IF [NOT] {cond} THEN] BREAK
    [IF [NOT] {cond} THEN] CONTINUE
LOOP
```

Notes: Nested WHILE & IF structures supported.

BREAK aborts WHILE/LOOP processing.

CONTINUE resumes processing at the top of WHILE/LOOP.

Multiple AND|OR logic (cannot mix AND/OR).

Dynamic Variables and/or Environment Variables can be used as {cond} variables.

Hint: Exercise caution to insure that the WHILE / LOOP processing relinquishes CPU time periodically. This can be accomplished with either the SLEEP or WAIT commands. A continuous and uninterrupted WHILE / LOOP will adversely affect CPU utilization.

Examples:

```
// Define %0 to 0 & use loop to add 1 until %0==10
DEFINE %0 00
WHILE
    DEFINE %0 %0+=01
    IF %0==10 THEN BREAK
LOOP

// Now loop to sub 1 from %0 (10) until %0==00
WHILE %0>00
    DEFINE %0 %0-=01
LOOP

// Wait for another task to complete before resuming
WHILE ACTIVE_TASK “THEOTHER.TSK”
    SLEEP 1
LOOP

// Check every second until file is closed (i.e., not in use)
WHILE FILE_IN_USE APPS:\DB\CUST.DB
    SLEEP 1
LOOP

// Wait for the message to appear on CURRENT_SCREEN
WHILE NOT SCAN_SCREEN “Backup finished!”
    SLEEP 1
LOOP
```

Conditional Tests

The following Conditional Tests can be used within a Task file that will be processed by the TaskMaster / TaskMaster Lite (TMLite) enhanced Batch Processor. Conditional Tests are used in conjunction with Conditional Structures (IF/WHILE). The tests return TRUE or FALSE conditions which can be used to direct processing.

== (EQUAL)	ACTIVE_TASK	LOGGED_IN
> (GREATER)	CURRENT_SCREEN	MOUNTED
< (LESS)	ERRORLEVEL	SCAN_FILE
>= (GREATER OR EQUAL)	EXIST	SCAN_SCREEN
<= (LESS OR EQUAL)	FILE_IN_USE	SCAN_STRING
	LOADED	SCREEN_LOCKED

Note: When using comparison tests (==, >, <, >=, or <=), spaces are not allowed between the strings or variables to be compared and the operators (signs). Quotes (“”) must be used to enclose strings which have the potential to be NULL or which may contain spaces within the text.

Examples:

"TESTTEXT"=="TESTTEXT"	(valid)
"Test Text"=="Test Text"	(valid, quote enclosed text)
“”==”“	(valid, quote enclosed [NULL] text)
Test Text==Test Text	(invalid, space in text - no quotes)
"TESTTEXT"== "TESTTEXT"	(invalid, space after == sign)
"Test Text" == "Test Text"	(invalid, space before/after == sign)

Any of the Dynamic Variables or Environment Variables may be used within a comparison test string. The batch processor will automatically substitute the appropriate value before performing the evaluation.

Examples:

"%SCREEN_NAME%"=="System Console"	(valid)
%CPU_UTIL%==100	(valid)

== (EQUAL)

ASCII hex compare by character and for length.

Examples:

True:	123==123	"ABC"=="ABC"	XYZ==XYZ
False:	321==123	"ABCD"=="ABC"	xyz==XYZ

> (GREATER)

ASCII hex compare by character and for length.

Examples:

True:	2>1	1>02	"ABCD">"ABC"	abc>ABC
False:	-1>-2	-02>-1	"ABCD">"abc"	XYZ>abc

< (LESS)

ASCII hex compare by character and for length.

Examples:

True:	1<2	02<1	"ABC"<"ABCD"	XYZ<xyz
False:	-2<-1	-1<-02	"abc"<"ABCD"	abc<XYZ

>= (GREATER OR EQUAL)

ASCII hex compare by character and for length.

(GREATER and EQUAL examples apply.)

<= (LESS OR EQUAL)

ASCII hex compare by character and for length.

(LESS and EQUAL examples apply.)

ACTIVE_TASK

Checks for active (running) tasks matching the specification. This test utilizes the following format:

ACTIVE_TASK [server/]{task[.ext]}

Notes: {task} requires a DOS 8.3 file format specification. [server/] is optional (local Server is assumed if not specified). [.ext] is optional as both .NCF and .TSK are checked, if no extension is specified. Wild cards supported.

The current task is excluded from the test results.

Examples:

```
IF ACTIVE_TASK %TASK%
    ECHO Another copy of this same task is running on this Server...
    EXIT
ENDIF

IF ACTIVE_TASK TEST.TSK THEN ECHO TEST.TSK is running on this Server...

IF ACTIVE_TASK RSERVER/TEST.* THEN ECHO One or more TEST.* tasks are running on RSERVER...
```

CONSOLE_LOCKED

Verifies the lock status of the actively displayed Server Console screen.

Note: Locked screens (e.g., MONITOR / SCRSAVER security screens) cannot be changed without first being unlocked.

Check the current Console Screen using the conditional test: CONSOLE_SCREEN

Change the current Console Screen using the batch command: CONSOLE SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

Example:

```
IF CONSOLE_LOCKED
    ECHO Cannot change Console Screen until it is unlocked!
ELSE
    CONSOLE SCREEN “System Console”
ENDIF
```

CONSOLE_SCREEN

Checks if the name of the actively displayed Server Console screen matches "screen" (case sensitive match up to length of "screen"). This test utilizes the following format:

CONSOLE_SCREEN {"screen"}

Notes: The “screen” name must be enclosed in double quotes (“”).

The defined name for the actively displayed screen on a NetWare Server can be determined by pressing the Alt key on the Server keyboard. The active screen name should appear across the upper left hand corner.

Change the current Console Screen using the batch command: CONSOLE SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

Example:

```
IF CONSOLE_SCREEN “System Console”
    ECHO Console Screen is already: System Console
ELSE
    CONSOLE SCREEN “System Console”
ENDIF
```

CURRENT_SCREEN

Checks if the name of the current screen against which the active task performs keyin / screen interrogation commands matches "screen" (case sensitive match up to length of "screen"). This test utilizes the following format:

```
CURRENT_SCREEN {"screen"}
```

Notes: The “screen” name must be enclosed in double quotes (“”).

The defined screen name for the actively displayed screen on a NetWare Server can be determined by pressing the Alt key on the Server keyboard. The active screen name should appear across the upper left hand corner.

Change the current screen using the batch command: CONSOLE SCREEN / CURRENT SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

Example:

```
IF CURRENT_SCREEN “TMConsole (Shell)”
    ECHO Current Screen for Task I/O is already: TMConsole (Shell)
ELSE
   CURRENT SCREEN “TMConsole (Shell)”
ENDIF
```

ERRORLEVEL

Returns the status of the last command executed by the task.

Note: Returns a TRUE or FALSE condition, not an error code.

EXIST

Confirms the existence of a file or directory matching the specification. This test utilizes the following format:

```
EXIST [server/]{[[drive:|vol:]path\]file}
```

Notes: {file} can be a fully qualified specification or relative to the Current Working Directory (CWD). [server/] is optional (local Server is assumed if not specified). DOS partition and non-DOS names supported. Wild cards supported.

If {file} has embedded spaces, the specification must be enclosed in double quotes (i.e., “SYS:\Dir 1\File 1.Ext”).

Examples:

IF EXIST SYS:SYSTEM\GRPWISE.NLM THEN ECHO GRPWISE.NLM found in SYS:SYSTEM on this Server...

IF EXIST “SYS:HOME\TOM\My Files” THEN ECHO “SYS:HOME\TOM\My Files” found on this Server...

IF EXIST RSERVER/“SYS:HOME\TOM\My Files” THEN ECHO “SYS:HOME\TOM\ My Files” found on RSERVER...

FILE_IN_USE

Checks if a file matching the specification is in use (open) on the Server. This test utilizes the following format:

FILE_IN_USE [server/]{[[vol:]path\]file}

Notes: {file} can be a fully qualified specification or relative to the Current Working Directory (CWD). [server/] is optional (local Server is assumed if not specified). Non-DOS names supported. Wild cards supported.

If {file} has embedded spaces, the specification must be enclosed in double quotes (i.e., “SYS:\Dir 1\File 1.Ext”).

Examples:

IF FILE_IN_USE DATA:Acct\PAYROLL.DAT THEN ECHO DATA:Acct\PAYROLL.DAT is in use on this Server...

IF FILE_IN_USE “DATA:Acct\Payroll Data” THEN ECHO “DATA:Acct\Payroll Data” is in use on this Server...

IF FILE_IN_USE RSERVER/“DATA:Acct\Payroll Data” THEN ECHO “DATA:Acct\Payroll Data” in use on RSERVER

LOADED

Checks if a module matching the specification is loaded on the Server. This test utilizes the following format:

LOADED [server/]{module[.ext]}

Note: [.ext] is optional as all valid module types are checked (i.e., CLIB matches CLIB.NLM and MPS14 matches MPS14.PSM), if no extension is specified. [server/] is optional (local Server is assumed if not specified).

Examples:

IF LOADED GRPWISE.NLM THEN ECHO GRPWISE.NLM is loaded on this Server...

IF LOADED RSERVER/GRPWISE.NLM THEN ECHO GRPWISE.NLM is loaded on RSERVER...

LOGGED_IN

Checks a user matching the specification is logged into the Server. This test utilizes the following format:

LOGGED_IN {user}

Notes: If {user} has embedded spaces, the specification must be enclosed in double quotes (i.e., “User Name”).

MOUNTED

Confirms that the specified volume is mounted on the Server. This test utilizes the following format:

MOUNTED [server/]{vol}

Note: [server/] is optional (local Server is assumed if not specified).

Examples:

IF MOUNTED DATA: THEN ECHO Volume DATA: is mounted on this Server...

IF MOUNTED RSERVER/DATA: THEN ECHO Volume DATA: is mounted on RSERVER...

SCAN_FILE

Scans files matching the file specification for the string specification. This test uses the following format:

SCAN_FILE {"string"} {[[vol:]path\]file}

Notes: {“string”} must be enclosed within double quotes (“”) and the search is case sensitive.

{file} can be a fully qualified specification or relative to the Current Working Directory (CWD). Non-DOS names supported. Wild cards supported.

If {file} has embedded spaces, the specification must be enclosed in double quotes (i.e., “SYS:\Dir 1\File 1.Ext”).

SCAN_SCREEN

Scans the current screen against which the active task performs keyin / screen interrogation commands for the string specification. This test utilizes the following format:

SCAN_SCREEN {"string"}

Notes: {“string”} must be enclosed within double quotes (“”) and the search is case sensitive.

Check the current task I/O screen using the conditional test: CURRENT_SCREEN

Change the current task I/O screen using the batch commands: CONSOLE SCREEN / CURRENT SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

SCAN_STRING

Scans the second string specification for the first string specification. This test utilizes the following format:

SCAN_STRING {"substring"} {"string"}

Note: Both strings must be enclosed within double quotes (“”) and the search is case sensitive.

SCREEN_LOCKED

Verifies the lock status of the actively displayed Server Console screen.

Note: Locked screens (e.g., MONITOR / SCRSAVER security screens) cannot be changed without first being unlocked.

Check the current Console Screen using the conditional test: CONSOLE_SCREEN

Change the current Console Screen using the batch command: CONSOLE SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

Example:

```
IF SCREEN_LOCKED
    ECHO Cannot change Console Screen until it is unlocked!
ELSE
    CONSOLE SCREEN “System Console”
ENDIF
```

Commands

The following Commands can be used within a Task file that will be processed by the TaskMaster / TaskMaster Lite (TMLite) enhanced Batch Processor. They are supplemental to the Extended Console Commands added by the Server Module and the basic NetWare Server System Console Commands.

ABORT

Terminates the current task (if no specification given) or the specified command or task. This command utilizes the following format:

ABORT
ABORT {command}
ABORT {task}

Notes: ABORT (alone, no command or task specified) terminates the current task being processed.

ABORT {command} terminates the specified TaskMaster Extended Console Command, if active. (Refer to the ABORT command in the Console Commands section of this manual.)

ABORT {task} terminates the first occurrence on any matching task currently being processed. When specifying {task}, use the DOS 8.3 file name without volume or path. (Refer to the ABORT command in the Console Commands section of this manual.)

ACTIVE_SCREEN

Note: Obsolete (supported for backwards compatibility). Replacement: CONSOLE_SCREEN

Changes and activates the screen that is actively displayed on the Server Console. This command utilizes the following format:

ACTIVE SCREEN ["screen_name"]

Notes: The name assigned to a NetWare Server screen can be viewed by pressing the Alt key on the Server keyboard while the screen is actively displayed. The active screen name should appear across the upper left hand corner.

If a specific screen name is desired, the specification must match the name of the desired screen (up to the length of the specification which should be enclosed in quotes).

If the intent is to merely change to the next screen (i.e., as if pressing Alt-ESC on the Server console), either provide a null screen name ("") or do not specify any screen name.

The ERRORLEVEL flag is set if “screen name” does not match any NLM screen or the Server Console is locked.

Also changes the current screen against which the active task performs keyin / screen interrogation commands (i.e., also performs a CURRENT SCREEN using the specified screen name).

For backward compatibility with previous versions, ACTIVE_SCREEN continues to be supported.

Examples:

ACTIVE SCREEN "System Console"
Change activate the current screen actively displayed on the Server Console to the “System Console” screen.

ACTIVE SCREEN "" - or - ACTIVE SCREEN
Change to and activate the next screen in the NLM screens list (same as pressing Alt-Esc on the Server keyboard).

CALC

Performs basic integer math calculations (+ Add, - Subtract, * Multiply, / Divide) with the result stored in the specified Dynamic Variable (%0 - %9). This command utilizes the following format:

CALC {%#} {equation}

Notes: Dynamic Variable must be specified as an argument to receive the result of the calculation.

Dynamic Variables and Environment Variables can be specified within the equation. All such variables are resolved prior to the calculation being performed.

The following mathematical operations are supported:

+ Addition
- Subtraction
* Multiplication
/ Division

The declared size of the destination argument (%#) is used for the result whenever possible (i.e., declaring the argument as a four digit value, DEFINE %0 0000, will cause any result with less than four digits to be zero-filled: 0000 - 9999).

Examples:

CALC %0 ((%1 + %2) * 100) / %3
Adds the values in %1 and %2; then multiplies the result by 100 and divides that result by the value in %3; then stores the result in argument %0.

CALC %1 %FILE_SIZE_SYS:VOL$LOG.ERR% / 1024
Divides the %FILE_SIZE_% value for the SYS:VOL$LOG.ERR file by 1024 to calculate the Kb size of the SYS: Volume Error Log file then stores the result in argument %1.

CALL

Executes the specified task as a called procedure. The calling task suspends processing until the called task completes. This command utilizes the following format:

CALL [[vol:]path\]{task}[.ext]

Note: The default search path and extension rules apply as if the Task were executed via the TMConsole or the System Console using the TMRUN command.

CHANGE SCREEN Note: Obsolete (supported for backwards compatibility). Replacement: CONSOLE_SCREEN

Changes and activates the screen that is actively displayed on the Server Console. This command utilizes the following format:

CHANGE SCREEN ["screen_name"]

Notes: The name assigned to a NetWare Server screen can be viewed by pressing the Alt key on the Server keyboard while the screen is actively displayed. The active screen name should appear across the upper left hand corner.

If a specific screen name is desired, the specification must match the name of the desired screen (up to the length of the specification which should be enclosed in quotes).

If the intent is to merely change to the next screen (i.e., as if pressing Alt-ESC on the Server console), either

provide a null screen name ("") or do not specify any screen name.

The ERRORLEVEL flag is set if “screen name” does not match any NLM screen or the Server Console is locked.

Also changes the current screen against which the active task performs keyin / screen interrogation commands (i.e., also performs a CURRENT SCREEN using the specified screen name).

For backward compatibility with previous versions, CHANGE_SCREEN continues to be supported.

Examples:

CHANGE SCREEN "System Console"
Change activate the current screen actively displayed on the Server Console to the “System Console” screen.

CHANGE SCREEN "" - or - CHANGE SCREEN
Change to and activate the next screen in the NLM screens list (same as pressing Alt-Esc on the Server keyboard).

CLEAR FILE

This command has been replaced by the FILE CLOSE Extended Console Command.

Notes: For backward compatibility with previous versions, CLEAR_FILE and CLEAR FILE continue to be supported.

CLEAR LOCK

This command has been replaced by the FILE UNLOCK Extended Console Command.

Notes: For backward compatibility with previous versions, CLEAR_LOCK and CLEAR LOCK continue to be supported.

CLOSE

Used to close previously opened OPEN READ / OPEN WRITE files. This command utilizes the following format:

CLOSE [READ|WRITE] [#n]

Notes: Close a specific file, all files of a specific access type, or all OPEN READ / OPEN WRITE files.

Examples:

CLOSE READ #n	Close OPEN READ #n file (n = 0 - 9).
CLOSE READ	Close all OPEN READ files.
CLOSE WRITE #n	Close OPEN READ #n file (n = 0 - 9).
CLOSE WRITE	Close all OPEN WRITE files.
CLOSE	Close all OPEN READ / WRITE files.

CONSOLE SCREEN

Changes and activates the screen that is actively displayed on the Server Console. This command utilizes the following format:

CONSOLE SCREEN ["screen_name"]

Notes: The name assigned to a NetWare Server screen can be viewed by pressing the Alt key on the Server keyboard while the screen is actively displayed. The active screen name should appear across the upper left hand corner.

If a specific screen name is desired, the specification must match the name of the desired screen (up to the length of the specification which should be enclosed in quotes).

If the intent is to merely change to the next screen (i.e., as if pressing Alt-ESC on the Server console), either provide a null screen name ("") or do not specify any screen name.

The ERRORLEVEL flag is set if “screen name” does not match any NLM screen or the Server Console is locked.

Also changes the current screen against which the active task performs keyin / screen interrogation commands (i.e., also performs a CURRENT SCREEN using the specified screen name).

Examples:

CONSOLE SCREEN "System Console"
Change activate the current screen actively displayed on the Server Console to the “System Console” screen.

CONSOLE SCREEN "" - or - CONSOLE SCREEN
Change to and activate the next screen in the NLM screens list (same as pressing Alt-Esc on the Server keyboard).

CURRENT SCREEN

Changes and activates the screen that is actively displayed on the Server Console. This command utilizes the following format:

CURRENT SCREEN ["screen_name"]

Notes: The name assigned to a NetWare Server screen can be viewed by pressing the Alt key on the Server keyboard while the screen is actively displayed. The active screen name should appear across the upper left hand corner.

If a specific screen name is desired, the specification must match the name of the desired screen (up to the length of the specification which should be enclosed in quotes).

If the intent is to merely change to the next screen (i.e., as if pressing Alt-ESC on the Server console), either provide a null screen name ("") or do not specify any screen name.

Does not change the screen displayed on the Server Console nor activate the Server Console displayed screen.

The ERRORLEVEL flag is set if “screen name” does not match any NLM screen.

Examples:

CURRENT SCREEN "TMConsole (Shell)"
Change the current screen against which the active task performs keyin / screen interrogation commands to the “TMConsole (Shell)” screen.

CURRENT SCREEN "" - or - CURRENT SCREEN
Change the current screen against which the active task performs keyin / screen interrogation commands to the next screen in the NLM screens list.

DEBUG OFF / DEBUG ON

Start (DEBUG ON) / Stop (DEBUG OFF) the logging of batch file processing in the task log (same volume, path and file name as the executing task but with a .DBG extension).

DEBUG ON [opts]
DEBUG OFF

[opts]: TRUNC[ATE]
(Only one option may be specified)

TRUNC
TRUNCATE — Truncate the file, if it exists

Notes: DEBUG OFF is the default mode.

DEBUG ON should only be used as a debugging tool, not in normal operation. Each executed line in the batch file is written to the log and flushed to disk as processed. Use of DEBUG ON will impact Server overhead due to the added number of additional disk write operations required.

DEFINE

Used to assign contents to a Dynamic Variable. Like Environment Variables, Dynamic Variables can be used almost anywhere in a task, either as supplied parameters or as command operators. This command utilizes the following format:

DEFINE {%#} [string]

Notes: Dynamic Variables must be defined before being referenced. By default, they are defined as NULL strings.

The DEFINE declaration should be provided EXACTLY as required for the parameter or operator it is intended to replace. If quotes ("") are required, they should be included in the DEFINE string.

CAUTION should be exercised since the entire string will be used in the declaration.

If the string assigned to a Dynamic Variable is in valid numeric format, the value can be dynamically incremented and/or decremented as follows:

DEFINE %0 %0+=1 — increment the value by 1
DEFINE %0 %0-=1 — decrement the value by 1

The Dynamic Variable must be defined prior to being incremented/decremented and should be pre-formatted to the correct length/size for any comparisons:

DEFINE %0 000
3 digit fixed length format (-999 - 999)

DEFINE %0 0000
4 digit fixed length format (-9999 - 9999)

Examples:

DEFINE %0 Hello World!
'Hello World!' is stored in Dynamic Variable %0 (excluding the apostrophes surrounding the string).

DEFINE %1 %CPU_UTIL%
The current value for Environment Variable %CPU_UTIL% (3 digit numeric - 000-100) is stored in %1.

DEFINE %2 0000
The 4 digit fixed length numeric string 0000 is stored in %2.

DEFINE %3 %3+=1
Increment the value of the numeric string stored in %3 by 1 then store the result as a string in %3.

DEFINE %5 %4-=2
Decrement the value of the numeric string stored in %4 by 2 then store the result as a string in %5.

DELAY

Silently suspend batch file processing for the specified number of milliseconds. This command utilizes the following format:

DELAY ## (where ## is the number of milliseconds to sleep)

Note: Processing suspends silently (no message displayed). Supported interval of 50ms (min) to 4294967295ms (max).

ECHO

Displays any trailing comments on the TMConsole (Shell) screen. This command utilizes the following format:

ECHO {string}

Notes: If a null string or a period is placed immediately following the ECHO command, a blank line is displayed.

The contents of any Dynamics Variable(s) and/or Environment Variable(s) are automatically substituted.

Examples:

ECHO.	a null line
ECHO Hi	Hi
ECHO CPU Util: %CPU_UTIL%%	current CPU utilization

ECHO OFF / ECHO ON

Start (ECHO ON) / Stop (ECHO OFF) the display on the active batch screen of the task commands as they are processed. ECHO ON will display all of the commands while ECHO OFF suppresses all but required or command generated output.

Note: ECHO OFF is the default mode.

ECHO_WRITE

Displays the trailing comments on the active batch screen and also writes them to the corresponding OPEN WRITE #n (#0 - #9) file. This command utilizes the following format:

ECHO_WRITE #n {string}

Notes: Same as executing both an ECHO and WRITE #n command for the same string.

If a null string or a period is placed immediately following the ECHO_WRITE #n command, a blank line is displayed and written to the OPEN WRITE #n file.

The contents of any Dynamic Variable(s) and/or Environment Variable(s) are automatically substituted.

EXIT

Exits (silently terminates) the currently executing task.

FINDCHR

Finds the offset of the first occurrence of the specified character within the specified string and sets the specified Dynamic Variable accordingly. This command utilizes the following format:

FINDCHR %# "." "string" [CASE | LAST]

Notes: A Dynamic Variable must be specified as the argument to receive the numeric offset (destination - first specification) while a Dynamic Variable, Environment Variable, or text string (constant) may be specified as either the character argument to be searched for (second specification) or the string argument to be searched (third specification).

By default, the search is not case sensitive (i.e., upper and lower case versions of the same alpha character are considered to match). Appending the CASE option after the specifications will result in a case sensitive search.

By default, the offset (1 - ???) of the first character of the first matching occurrence is returned (offset within the string, excluding the double quotes enclosing the string. Appending the LAST option after the specifications will result in the offset of the first character of the last matching occurrence being returned (if LAST is specified and only one match is found, it is returned since it is also the last).

If the second specification (the character to search for) is a string (i.e., has more than one character), only the first character is used.

If either the second specification (the character to search for) or third specification (the string to be searched) is a text string or an Environment Variable, it must be enclosed in double quotes. If a Dynamic Variable is specified, the contents must begin and end with double quotes or the Dynamic Variable be enclosed in double quotes.

If either the second or third specification is null or no match is found, the destination Dynamic Variable is set to 0.

Example:

FINDCHR %0 "a" "ABCDEFabcdef"
%0 is set to 1 since the search is not case sensitive.

FINDCHR %1 "d" "ABCDEFabcdef" CASE
%1 is set to 10 since the search is case sensitive.

FINDCHR %1 "%2" "%3"
Assuming %2 is defined as 'A' and %3 is defined as 'Match Atlas' then %1 is set to 2 (not case sensitive). If the CASE option were specified then %1 is set to 7 (case sensitive).

FINDCHR %1 "%2" "%3" LAST
Assuming %2 is defined as 'A' and %3 is defined as 'Match Atlas' then %1 is set to 10 (last 'a' in the string). If the CASE option were specified then %1 is set to 7 (case sensitive, first and last occurrence of 'A').

Note: Double quotes enclosing the searched specification are not included in the search or the returned offset.

FINDLEN

Finds the specified string length and sets the specified Dynamic Variable accordingly. This command utilizes the following format:

FINDLEN %# "string"

Notes: A Dynamic Variable must be specified as the argument to receive the string length(destination - first specification) while a Dynamic Variable, Environment Variable, or text string (constant) may be specified as the string argument to measured (second specification).

If the second specification (the string to be counted) is a text string or an Environment Variable, it must be enclosed in double quotes (""). If a Dynamic Variable is specified, the contents must begin and end with double quotes or the Dynamic Variable be enclosed in double quotes. The length will not include the double quotes.

If the second specification is null or no match is found, the destination Dynamic Variable is set to zero.

Example:

FINDLEN %0 "ABCDEFabcdef"
%0 is set to 12.

FINDLEN %1 "%2"
Assuming %2 is defined as 'The End' then %1 is set to 7.

Note: Double quotes enclosing the specification are not included in the returned length.

FINDSTR

Finds the offset of the first occurrence of the specified substring in the specified string and sets the specified Dynamic Variable accordingly. This command utilizes the following format:

FINDCHR %# "sub" "string" [CASE | LAST]

Notes: A Dynamic Variable must be specified as the argument to receive the numeric offset (destination - first specification) while a Dynamic Variable, Environment Variable, or text string (constant) may be specified as either the substring argument to be searched for (second specification) or the string argument to be searched (third specification).

By default, the search is not case sensitive (i.e., upper and lower case versions of the same alpha character are considered to match). Appending the CASE option after the specifications will result in a case sensitive search.

By default, the offset (1 - ???) of the first character of the first matching occurrence is returned (offset within the string, excluding the double quotes enclosing the string). Appending the LAST option after the specifications will result in the offset of the first character of the last matching occurrence being returned (if LAST is specified and only one match is found, it is returned since it is also the last).

The second specification (the substring to search for) must exist in the same character order within the third specification.

If either the second specification (the string to search for) or third specification (the string to be searched) is a text string or an Environment Variable, it must be enclosed in double quotes. If a Dynamic Variable is specified, the contents must begin and end with double quotes or the Dynamic Variable be enclosed in double quotes.

If either the second or third specification is null or no match is found, the destination Dynamic Variable is set to 0.

Example:

FINDSTR %0 “abc” “ABCDEFabcdef”
%0 is set to 1 since the search is not case sensitive.

FINDSTR %1 “def” “ABCDEFabcdef” CASE
%1 is set to 10 since the search is case sensitive.

FINDSTR %1 “%2” “%3”
Assuming %2 is defined as ‘At’ and %3 is defined as ‘Match Atlas’ then %1 is set to 2 (not case sensitive). If the CASE option were specified then %1 is set to 7 (case sensitive).

FINDSTR %1 “%2” “%3” LAST
Assuming %2 is defined as ‘A’ and %3 is defined as ‘Match Atlas’ then %1 is set to 10 (last ‘a’ in the string). If the CASE option were specified then %1 is set to 7 (case sensitive, first and last occurrence of ‘A’).

Note: Double quotes enclosing the searched specification are not included in the search or the returned offset.

GOTO

Moves the batch execution to the line that corresponds to the label specified after the GOTO command. Any line beginning with a colon (:) is regarded as an execution label for subsequent GOTO redirection. This command utilizes the following format:

GOTO label

Notes: Using GOTO within IF / ENDIF and WHILE / LOOP conditional structure is not advised as jumping outside of the construct is likely to cause an IF / ENDIF or WHILE / LOOP count mismatch error to occur and can have indeterminate (negative) impact on the process stack.

It is important to differentiate between an IF / ENDIF conditional structure and an IF THEN conditional test. GOTO can be safely used with an IF THEN conditional test since it only executes the single command.

Example:

GOTO START
Shift execution to the line following the label :START in the task.

IF [conditional_test] THEN GOTO NEXT_SECTION
Shift execution to the line following the label :NEXT_SECTION in the task IF the conditional test is TRUE.

KEYIN

Used to enter string and/or special keys into the current screen against which the active task performs keyin / screen interrogation commands as if input from the Server's keyboard. This command utilizes the following format:

KEYIN {keyword}
KEYIN {"string"}
KEYIN {keyword|"string"} [{keyword|"string"}]...

Notes: Multiple “string” and/or keyword combinations may be combined in a single KEYIN command.

A string may contain one or more alphanumeric characters and must be enclosed within double quotes ("").

Check the current task I/O screen using the conditional test: CURRENT_SCREEN

Change the current task I/O screen using the batch commands: CONSOLE SCREEN / CURRENT SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

Keyword list:

BACKSPACE, ENTER, ESC, ESCAPE	
F1 ... F12	
UP, DOWN, LEFT, RIGHT	
PGUP, PGDN, HOME, END	
INS, INSERT, DEL, DELETE	
SHIFT-keyword	(hyphen/dash required)
ALT-keyword	(hyphen/dash required)
CTRL-keyword	(hyphen/dash required)
ALT-A ... ALT-Z	(Alt-[char] keystrokes)
CTRL-A ... CTRL-Z	(Ctrl-[char] keystrokes)

LOG OFF / LOG ON Note: Obsolete (supported for backwards compatibility). Replacement: DEBUG OFF / DEBUG ON

Start (LOG ON) / Stop (LOG OFF) the logging of batch file processing in the task log (same volume, path and file name as the executing task but with a .DBG extension).

LOG ON [opts]
LOGG OFF

[opts]: TRUNC[ATE]
(Only one option may be specified)

TRUNC	
TRUNCATE	Truncate the file, if it exists

Notes: LOG OFF is the default mode.

LOG ON should only be used as a debugging tool, not in normal operation. Each executed line in the batch file is written to the log and flushed to disk as processed. Use of LOG ON will impact Server overhead due to the added number of additional disk write operations required.

OPEN READ

Open the specified file for sequential reading of data using the READ command. This command utilizes the following format:

OPEN_READ #n [[vol:]path\]{spec}

Notes: A file handle (#0 - #9) must be specified. If the file handle is actively in use for another OPEN READ file, the previously open file is closed prior to the new file being opened (i.e., only one file can be open concurrently for each file handle).

The ERRORLEVEL flag is set TRUE if the specified file cannot be opened for read access.

{spec} can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD).

For backward compatibility with previous versions, OPEN_READ without a file handle specification continues to be supported and defaults to file handle #0.

OPEN WRITE

Open the specified file for sequential output of data using the WRITE command. This command utilizes the following format:

OPEN_WRITE #n [[vol:]path\]spec [opts]

[opts]: TRUNC[ATE]
(Only one option may be specified)

TRUNC TRUNCATE	Truncate the file, if it exists

Notes: A file handle (#0 - #9) must be specified. If the file handle is actively in use for another OPEN READ file, the previously open file is closed prior to the new file being opened (i.e., only one file can be open concurrently for each file handle).

If the file exists and TRUNC or TRUNCATE is not specified, the file will be opened in append mode.

The ERRORLEVEL flag is set TRUE if the specified file cannot be opened for write access.

{spec} can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD).

For backward compatibility with previous versions, OPEN_WRITE without a file handle specification continues to be supported and defaults to file handle #0.

PARSE

Extract data from a Dynamic Variable, Environment Variable, or text string (constant) into a Dynamic Variable. This command utilizes the following format:

PARSE %# %# [#-#,#-#,...] [APPEND | PACK]

Notes: A Dynamic Variable must be specified as the argument to receive the extracted data (destination - first specification) while a Dynamic Variable, Environment Variable, or text string (constant) may be specified as the argument supplying the data (source - second specification).

Unless one or more key patterns are specified, the data is transferred exactly as it exists in the source variable.

Up to ten key patterns ([#-#,#-#,...]) can be specified to extract particular data from the source and to organize the result into a distinct format. Key patterns utilize a beginning position and ending position format separated by a hyphen, with each key pattern separated by a comma.

By default, if a key pattern exceeds the data length of the source variable (i.e., 1-100 is specified but the data in the source is only 80 bytes long), the data returned for the key pattern will be padded with spaces (a full 100 bytes are returned). This padding feature can be used to format the data since specifying a key pattern which is known to be longer than that defined or allowed in a variable will return space filled data equal to the key pattern.

Any options specified must follow any key pattern specifications. The PACK option will result in the truncation of shorter data to its actual length (i.e., a key pattern of 1-100 is specified but the record is only 80 bytes long will result in only 80 bytes being returned). By default, parsed data overwrites the existing contents of the destination variable unless the APPEND option is specified in which case the new data follows any existing data.

Data that is longer than the maximum amount supported by a Dynamic Variable is truncated.

Examples:

PARSE %0 %1
Copy the defined data in %1 to %0. (Same as DEFINE %0 %1)

PARSE %1 %2 12,1-10,28-127
Extract the data at byte 12, bytes 1-10, and bytes 28-127 defined in %2 then store the extracted data in the specified order in %1. If the defined data in %2 is shorter than any key ending position, the data returned for the key pattern will be padded with spaces to the full key length.

PARSE %2 %3 1-10,512-516,11-20
Extract the data at bytes 1-10 and bytes 11-20 from %3. If the key pattern 512 - 516 happens to be greater than the maximum size supported by the Dynamic Variable source, space filled data will be substituted and combined with the extracted data then stored in the specified order in the Dynamic Variable %2. If the source data is shorter than any other key ending position, the data returned for the key pattern will be padded with spaces to the full key length.

PARSE %3 %4 10-10,1-9,11-127 PACK
Extract the data at byte 10, bytes 1-9, and bytes 11-127 from %4 then store the extracted data in the specified order in the Dynamic Variable %3. If the defined data in %4 is shorter than any key ending position, the data returned for the key pattern will be truncated.

PARSE %3 %4 10-10,1-9,11-127 APPEND PACK
Extract the data at byte 10, bytes 1-9, and bytes 11-127 from %4 then store the extracted data in the specified order in the Dynamic Variable %3 after any data which already exists in %3. If the defined data in %4 is shorter than any key ending position, the data returned for the key pattern will be truncated.

PAUSE

Suspend processing until a key is pressed on the Server's keyboard or the specified timeout period occurs (in seconds). The TMConsole (Shell) screen will be activated and will emit an audible beep once per second until a key is pressed or the specified timeout period passes. This command utilizes the following format:

PAUSE [##]

Note: Specifying a numeric value command indicates a maximum period of time (in seconds) to pause the processing. If the timeout period passes without a key being pressed, the ERRORLEVEL flag is set and processing continues.

READ

Reads the next sequential record from a previously opened OPEN_READ file and stores the data in the specified Dynamic Variable. This command utilizes the following format:

READ #n %# [#-#,#-#,...] [%# [#-#,#-#,...]] [PACK]

Notes: A file handle (#0 - #9) which corresponds to a successfully processed OPEN READ #n command must be specified.

At least one Dynamic Variable must be specified to receive the read data (the destination variable).

The ERRORLEVEL flag is set TRUE if an error occurs during the read or if the End Of File (EOF) is reached.

Unless one or more data key patterns have been specified, the data is stored in the specified variable as it is read from the file, up to the carriage return / line feed (End Of Record) or the maximum amount of data supported by a Dynamic Variable has been read (whichever occurs first).

Up to ten key patterns ([#-#,#-#,...]) can be specified to extract select data from each record and to organize the result into a distinct format before it is stored into the associated Dynamic Variable. Key patterns utilize a beginning position and ending position format separated by a hyphen, with each key pattern separated by a comma. If no ending position is specified, the data selected is assumed to contain only the one byte located at the beginning position.

The same data record can be read and parsed into multiple Dynamic Variables, even with overlapping data or key patterns, by specifying more than one Dynamic Variable and key specifications on the same READ command line. However, the data record to be read and then parsed into the Dynamic Variable(s) is limited to 4095 bytes total. Data beyond the 4095 limitation is ignored and the file pointer is positioned at the next sequential record.

By default, if a key pattern exceeds the record length (i.e., a key pattern of 1-100 is specified but the record is only 80 bytes long), the data returned for the key pattern beyond the End Of Record (bytes 81 through 100) will be padded with spaces (a full 100 bytes are returned). This padding feature can be used to format the data since specifying a key pattern which is known to be longer than the record length will return space filled data equal to the key pattern.

Appending the PACK option after key pattern specifications will result in the truncation of shorter data to its actual length (i.e., a key pattern of 1-100 with a record containing only 80 bytes results in only 80 bytes being returned).

Any data that is longer than the maximum amount supported by a Dynamic Variable is truncated and the file pointer is positioned at the next sequential record.

For backward compatibility with previous versions, READ without a file handle specification continues to be supported and defaults to file handle #0.

Examples:

READ %0
Read the next record, placing up to the first 511 bytes of the data in the Defined Variable%0.

READ %1 12,1-10,28-127
Read the next record, extracting the data at byte 12, bytes 1-10, and bytes 28-127 then store the extracted data in the order specified in the Defined Variable %1. If the record is shorter than any key ending position, the data returned for the key pattern will be padded with spaces to the full key length.

READ %2 1-10,512-516,11-20
Read the next record, extracting the data at bytes 1-10, bytes 512-516, and bytes 11-20 then store the extracted data in the order specified in Defined Variable %2. If the record is shorter than any key ending position, the data returned for the key pattern will be padded with spaces to the full key length. In this example, the second key pattern (512-516) could be used to separate the surrounding key patterns (1-10 and 11-20) for records that are shorter than 512 bytes.

READ %2 1-10,21-30,41-50 PACK %3 11-20,31-40,51-60
Read the next record and extract the data in bytes 1-10, 21-30, and 41-50 then store the extracted data in Defined Variable %2, truncating the data if the record is shorter than any key ending position. Re-using the same record data, extract the data in bytes 11-20, 31-40, and 51-60 then store the extracted data in Defined Variable %3, padding the data if the record is shorter than any key ending position.

REFORMAT

Modifies the format of the data stored in an Aliased or Defined Variable. This command utilizes the following format:

REFORMAT %# [opts]

[opts]: LEFT PACK RIGHT TEXT ##

LEFT	Left justify data (i.e., shift data left to replace any leading spaces).
PACK	If numeric data, strip any leading zeroes leaving only actual number (unless TEXT also specified). If alphanumeric and LEFT or RIGHT is not specified, strip any leading and trailing spaces in the data.
RIGHT	Right justify data (i.e., shift data right to replace any trailing spaces).
TEXT	Retains leading zeroes on numeric data (only applies when PACK option also specified).
##	Reset data length to ## (truncate if longer else add trailing {default / LEFT} or leading {RIGHT} spaces)

Note: More than one option can be specified. However, specifying both LEFT and RIGHT will result in an error.

Examples:

REFORMAT %1 LEFT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘000003 ‘
REFORMAT %1 PACK	Before: %1 = ‘ 000003 ‘	After: %1 = ‘3'
REFORMAT %1 PACK TEXT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘000003'
REFORMAT %1 RIGHT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘ 000003‘
REFORMAT %1 LEFT PACK	Before: %1 = ‘ 000003 ‘	After: %1 = ‘3 ‘
REFORMAT %1 LEFT PACK TEXT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘000003 ‘
REFORMAT %1 PACK RIGHT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘ 3‘
REFORMAT %1 PACK RIGHT TEXT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘ 000003‘
REFORMAT %1 8	Before: %1 = ‘ 000003 ‘	After: %1 = ‘ 000003‘
REFORMAT %1 8 LEFT PACK	Before: %1 = ‘ 000003 ‘	After: %1 = ‘3 ‘
REFORMAT %1 8 LEFT PACK TEXT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘000003 ‘
REFORMAT %1 8 PACK	Before: %1 = ‘ 000003 ‘	After: %1 = ‘3 ‘
REFORMAT %1 8 PACK TEXT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘000003 ‘
REFORMAT %1 8 PACK RIGHT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘ 3‘
REFORMAT %1 8 PACK RIGHT TEXT	Before: %1 = ‘ 000003 ‘	After: %1 = ‘ 000003‘

REPLACE

Searches for the first character in each pair and replaces every occurrence found in the specified Dynamic Variable with the second character in the pair. This command utilizes the following format:

REPLACE %# "??"

Notes: A Dynamic Variable must be specified as the source of the data to be modified.

The replacement string must contain one or more pairs of characters. REPLACE will search for every occurrence of the first character in each pair and replace it with the second character in the pair.

Examples:

REPLACE %1 "_-"
Replace every underscore (_) in %1 with a hyphen (-).
Before = 'SERVER_411' After = 'SERVER-411'

REPLACE %2 " ._-Zz"
Replace every space ( ) in %2 with a period (.); Then replace every underscore (_) in %2 with a hyphen (-); Finally, replace every uppercase Z in %2 with a lowercase z.
Before = 'SERVER_411 Zero' After = 'SERVER-411.zero'

REWIND

Reset the sequential file record pointer to the beginning (start) of the file. This command utilizes the following format:

REWIND READ [#n]
REWIND WRITE [#n]

Notes: If no file handle (#0 - #9) is specified, all active file handles of the specified type (READ or WRITE) are reset.

REWIND is the equivalent of closing then re-opening the file, but without the overhead of those extra operations.

The ERRORLEVEL flag is set TRUE if any file(s) cannot reset.

{spec} can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD).

SAVE SCREEN

Captures the text image of the the current screen against which the active task performs keyin / screen interrogation commands to the specified file. This command utilizes the following format:

SAVE SCREEN [[vol:]path\]spec [opts]

[opts]: /BIN /BINARY /TEXT /TXT
(Only one option may be specified)

/BIN
/BINARY Stores the image in binary format
(character and attribute)
/TXT
/TEXT Stores the image in text format
(character only - default)

Note: {spec} can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD).

Check the current task I/O screen using the conditional test: CURRENT_SCREEN

Change the current task I/O screen using the batch commands: CONSOLE SCREEN / CURRENT SCREEN

Supports only NetWare character based screens, not NetWare v5 (or later) GUI (graphical) screens.

For backward compatibility with previous versions, SAVE_SCREEN continues to be supported.

SHIFT

Rotate the contents of the default Dynamic Variables (%0 through %9) left one position then clear User Define Variable (%9).

Note: The contents of %1 are moved to %0, %2 to %1, %3 to %2, and so forth with %9 set to null.

SHUTDOWN

Equivalent to a NetWare DOWN command entered at the Server' System Console with the added feature that it will automatically close any open files without a prompt for operator intervention.

SLEEP

Silently suspend batch file processing for the specified number of seconds. This command utilizes the following format:

SLEEP ## (where ## is the number of seconds to sleep)

Note: Processing suspends silently (no message displayed).

SYSENV

Retrieves the contents of or stores a value to a specified NetWare System Environment variable. This command utilizes the following format:

SYSENV GET {Dynamic Variable} {string}
SYSENV PUT {string}

Note: GET retrieves the contents of the NetWare System Environment variable matching {string} which it then stores in (Dynamic Variable}. GET {string} should be the desired NetWare System Environment variable, excluding the trailing equal sign (i.e., PATH not PATH=).

PUT stores {string} in the NetWare System Environment table. {string} should be the full variable to be stored in the NetWare System Environment table (i.e., PATH=SYS:\SYSTEM).

NetWare System Environment variables and TaskMaster Environment Variables are separate entities. They are not interchangable even if they share the same name.

Note: NetWare uses a single preceding % as its variable designator while TaskMaster encloses the variable name with a % (i.e., one on each end).

TOCALENDAR

Converts numeric day number within year (%NDAY_OF_YEAR%) to corresponding numeric calendar date format. This command utilizes the following format:

TOCALENDAR %# {%# | d | dd | ddd | yyyyd | yyyydd | yyyyddd}

Note: A Dynamic Variable must be specified as the destination for the new numeric date data.

The day number within the current year can be represented as a single to three digit number (1-366 or 001-366). Unless a year is specified (20031 - 2003365 or 2004001 - 2004366), the current year is assumed.

Examples:

```
TOCALENDAR %1 1                          After: %1 = 20040101 (during 2004)
TOCALENDAR %1 60                         After: %1 = 20040229 (during 2004)
TOCALENDAR %1 106                        After: %1 = 20040415 (during 2004)
TOCALENDAR %1 20031                      After: %1 = 20030101
TOCALENDAR %1 200331                     After: %1 = 20030131
TOCALENDAR %1 2003365                    After: %1 = 20031231

DEFINE %2 %NDAYOFYEAR%+=7
TOCALENDAR %1 %2                         After: %1 = (yyyymmdd - 7 days after today)

DEFINE %2 %NDAYOFYEAR%-=7
TOCALENDAR %1 %2                         After: %1 = (yyyymmdd - 7 days before today)
```

TODAYOFYEAR

Converts numeric calendar date (yyyymmdd or mmdd) to corresponding numeric day number within the specified or default year (%NDAY_OF_YEAR%). This command utilizes the following format:

```
TODAYOFYEAR %# {%# | mmdd | yyyymmdd}
```

Note: A Dynamic Variable must be specified as the destination for the new numeric date data.

The calendar date must be provided in the form yyyymmdd (20040615) or mmdd (0615). If the year is not specified, the current year is used for the conversion (normally not an issue except after February in leap years).

Examples:

```
TODAYOFYEAR %1 0101                      After: %1 = 001
TODAYOFYEAR %1 20030301                  After: %1 = 060
TODAYOFYEAR %1 20040301                  After: %1 = 061 (2004 was a leap year)

DEFINE %2 %FILE_UPDATE_SYS:\VOL$LOG.ERR%
TODAYOFYEAR %1 %2                        After: %1 = (SYS:VOL$LOG.ERR Last Modified day of year)
```

TOLOWER

Converts all alphabetic characters in the specified Dynamic Variable to lower case. This command utilizes the following format:

```
TOLOWER %#
```

Note: A Dynamic Variable must be specified as the source of the data to be modified.

Example:

```
TOLOWER %1
Before = 'ToLower'          After = 'tolower'
```

TOUPPER

Converts all alphabetic characters in the specified Dynamic Variable to upper case. This command utilizes the following format:

TOUPPER %#

Note: A Dynamic Variable must be specified as the source of the data to be modified.

Example:

TOUPPER %1
Before = 'ToUpper' After = 'TOUPPER'

VARALIAS

Used to change the default name or reference for a Dynamic Variable. Like Environment Variables, Dynamic Variables can be used almost anywhere in a task, either as supplied parameters or as command operators. This command utilizes the following format:

VARALIAS {%0-%9|%VAR00%-%VAR99%|%old_name%} {%new_name%}

Notes: %0 through %9 are reserved names for the first ten dynamic variables. %VAR00% through %VAR99% are default names assigned to the Dynamic Variable array. The contents of %0 and %VAR00% reference the first Dynamic Variable in the array. Likewise, %9 and %VAR09% reference the tenth Dynamic Variable in the array.

Using VARALIAS, it is possible to change the default name (%VAR00% through %VAR99%) used to reference a Dynamic Variable to something which better describes its usage. It is important to note that changing the default name for a Dynamic Variable does not change its array location or data contents, only the default name used to reference the information assigned to the Dynamic Variable. It should also be noted that changing the default name for any of the first 10 Dynamic Variables (%VAR00% through %VAR09%) does not have any affect on the reserved names %0 through %9 assigned to the same Dynamic Variables.

The first specification, the Dynamic Variable to be renamed, must be a valid Dynamic Variable name. With the exception of %0-%9 which are reserved names for the first ten dynamic variables and always exist, once a dynamic variable has been renamed, it can no longer be referenced by its former name, only the new name. However, the new name can then be renamed back to the original name or to a different name at which point the previous (renamed) name can no longer be referenced.

The naming rules for Alias Variables are as follows: The name must begin and end with a percent sign (%...%) and may contain a maximum of 15 characters (case insensitive), including the percent signs. The first character after the initial percent sign (%) many not be a numeric digit (i.e., %0...% through %9...% is not allowed). No spaces are allowed in the name (use dashes or underscores). Cannot match (duplicate) any other Alias Variable, Dynamic Variable or Environment Variable name.

Examples:

VARALIAS %VAR00% %ZERO%
Legal - %0 and %ZERO% now represent the same Dynamic Variable which was formerly represented by %VAR00%. %ZERO% replaces %VAR00% which is no longer a valid Dynamic Variable name. %0 remains valid (reserved).

VARALIAS %0 %ZERO%
Legal - Same as above (i.e., %ZERO% replaces %VAR00%, both representing the Dynamic Variable %0).

VARALIAS %VAR10% %Counter%
Legal - %Counter% (or %COUNTER%) replaces %VAR10% which is no longer a valid Dynamic Variable name.

VARALIAS %Count% %TotalCount%
Legal - %TotalCount% (or %TOTALCOUNT%) replaces %Count% which is no longer a valid Dynamic Variable name.

VARALIAS %TotalCount% %VAR10%
Legal - Reverse the previous changes (i.e., restore the original default Dynamic Variable name).

VARALIAS %VAR99% %Antidisestablishmentarianism%
Invalid - New name too long.

VARALIAS %VAR99% MyVar
Invalid - New name invalid/unsupported name format.

VARALIAS MyVar %OurVar%
Invalid - Dynamic Variable to be renamed is invalid (invalid/unsupported name format).

WAIT

Suspend the batch file processing for the specified number of seconds. This command utilizes the following format:

WAIT ## (where ## is the number of seconds to wait)

Note: Displays a message indicating the amount of time processing will be suspended.

WRITE

Writes any trailing comments to the specified OPEN WRITE file. This command utilizes the following format:

WRITE [-EOL] #n [string]

Notes: A file handle (#0 - #9) which corresponds to a successfully processed OPEN WRITE #n command must be specified.

The -EOL option suppresses the writing of the End Of Line (Carriage Return/Line Feed - CR/LF) line terminator allowing one or more WRITE commands to write data to a single line (record).

If a null string or a period is placed immediately following the file handle specification and the -EOL option is not specified, a blank line is written to the file.

The appropriate contents of any Dynamic Variables and/or Environment Variables are automatically substituted prior to being written.

The ERRORLEVEL flag is set TRUE if an error occurs.

For backward compatibility with previous versions, WRITE without a file handle specification continues to be supported and defaults to file handle #0.

Examples:

WRITE #0 Write what follows #0 to the OPEN WRITE #0 associated file with a CR/LF (EOL).

WRITE -EOL #1 Write what follows #1 to the OPEN WRITE #1associated file without an EOL (-EOL) and,
WRITE #1 since previous WRITE #1 did not include a CR/LF (-EOL), this becomes part of the same line (record).

WRITE #2 Date: %MONTH_NAME% %DAY%, %YEAR% / Time: %HOUR%:%MINUTE%%AM_PM%
(Write the current date and time {DATE: June 15, 2004 / Time: 6:17pm} to the OPEN WRITE #2 associated file.)

[INTENTIONAL BLANK PAGE]

Chapter 4 - Console Commands

The Server Module extends the basic Server System Console Commands which can be executed as part of a Task File; in .NCF files (NetWare Command File on NetWare Servers); or manually at the Server System Console. These Console Command extensions are divided into two categories: Local Server Commands which process on the local (host) Server; And, Remote Server Commands which interact with Remote Servers running a compatible version of the Server Module.

The following points should be noted:

- All commands are case insensitive, unless otherwise noted.

- With few exceptions, all of the commands (local and remote) must be executed via the TMConsole (Shell) prompt. The restriction is for security purposes and to avoid conflict with the Server System Console and its commands.

- Re-iterative commands (commands which are recursive or tend to display more than a single screen of information) create a separate screen to receive the output rather than clutter the TMConsole (Shell) or the Server System Console screens. The name of the screen (as viewed by pressing the ALT key when the screen is actively displayed) is set to the first part of the command line being processed.

  Once the command has completed processing, the closing of the created screen is based upon the following criteria:

  - If the command is executed manually via the TMConsole (Shell) without output redirection having been specified, the prompt <Press any key to close the screen> will appear for a period of time (user defined, default = 60 seconds) after which the window will automatically close even without operator intervention.

  - If the command is executed manually via the TMConsole (Shell) with output redirection specified, if the command is executed via a Task during batch processing, or if the command is executed via a Remote Server Command submission (TMSCMD), the window will automatically close without operator intervention.

  Examples of typically re-iterative commands include: CHMOD, CHOWN, COPY, DIR, DUMP, FIND, FLAG, FLAGDIR, GRANT, PURGE, REVOKE, SCOPY, SORT, SXCOPY, SYNC, TMUPDATE, TREE, TYPE, WHEREIS, WHOHAS, XCOPY, and similar commands.

- Most of the Console Commands extensions support output redirection allowing the information normally displayed on the screen to be redirected to a file. Where supported, output redirection is accomplished by appending the following to the command line with the results as described:

>filespec	Create/overwrite the output file
>>filespec	Append to an existing file

  Even when output redirection is used, re-iterative commands (described above) will still create their own screen so that they can display information as to the status of the processing. However, when output redirection is used, these screens will automatically close once the command completes its processing.

- Items separated by a vertical bar | indicate any one of the items may be used, but not more than one.

- Items in brackets [ ] are optional.

- Items in curly braces { } are required parameters.

Local Server Commands

The following Console Command extensions provided by the Server Module support Local Server operations.

Note: If the TaskMaster Secure NLM (TMSECURE.NLM) is loaded, most of these commands will only be supported when executed within a scheduled Task.

ABORT

Terminates the specified active task or supported Console Command extension. This command utilizes the following format:

```
ABORT {command}
ABORT SYNC [/#]
ABORT {task} [/#|/A]
```

[opts]:
/# Abort # (numeric) occurrence of the active {task} or SYNC job
/A Abort all matching {task} occurrences
(default = terminate first occurrence of {task} or all {command} instances)

Notes: ABORT {command} can be used to terminate most recursive Server Commands, including: CHMOD, CHOWN, COPY, DIR, DELTREE, FIND, FLAG, FLAGDIR, PURGE, SCOPY, SORT, SXCOPY, SYNC, TMUPDATE, TREE, WHEREIS, and XCOPY. ABORT {command} terminates all active instances of the {command}.

ABORT {task} terminates the first instance of any active task which matches the specification provided. To terminate a specific task, use the DOS 8.3 name (filename.ext or just filename) of an any active task for {task}. Non-DOS Name Space names are not supported but wild cards are supported.

The /# option can be used to match a specific instance of any active task which matches the specification provided (e.g., ABORT *.* /2 will terminate the second active task shown by the TASKLIST command). The /# option can also be used to match a specific active SYNC job (e.g., ABORT SYNC /2 will terminate the second SYNC job shown by the SYNC LIST command). The /A option can be used to terminate all instances of any active task which matches the specification provided (e.g., ABORT *.* /A terminates all active tasks while ABORT TEST.TSK terminates all active instances of the TEST.TSK task).

The task terminates once processing of the then active command line completes. If an SXCOPY or SYNC command line is active, it will be automatically aborted as part of the task termination. If any other recursive or repetitive command is active, the task will not terminate until the command completes, unless the active command is also terminated (must be among the commands supported by the ABORT {command} option, see above).

Examples:

ABORT TEST
will abort the first occurrence of any active task whose name matches TEST, TEST.NCF, or TEST.TSK

ABORT TEST.TSK
will abort the first occurrence of any active task whose name matches TEST.TSK

ABORT TEST*.*
will abort the first occurrence of any active task whose name matches TEST*.*

ABORT TEST.TSK /2
will abort the second occurrence of any active task whose name matches TEST.TSK

ABORT TEST.TSK /A
will abort all occurrences of any active task whose name matches TEST.TSK

ABORT TEST*.* /A
will abort all occurrences of any active task whose name matches TEST*.*

ABORT COPY
ABORT XCOPY
will terminate aall active COPY/XCOPY processing

ABORT SYNC
will terminate all active SYNC processes

ABORT SYNC /2
will terminate the second instance of active SYNC processing (as shown in SYNC LIST output)

Note: To ABORT a specific SYNC process when multiple, concurrent SYNC processes are active, first use the SYNC LIST command to identify the SYNC process to be terminated then use ABORT SYNC /# where # is the number of the SYNC process (as shown in the output of the SYNC LIST command) to be terminated.

APPEND

Appends the source file to the destination file. This command utilizes the following format:

APPEND [[d:]path\]{src} [[d:]path\]{dest}
APPEND [[vol:]path\]{src} [[vol:]path\]{dest}

Notes: Each specification can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD).

Support for the DOS partition requires that the specification includes a valid DOS drive and path.

CD / CHDIR

Changes the Current Working Directory (CWD). This command utilizes one of the following formats:

CD [vol:][path]
CHDIR [vol:][path]

Notes: [path] can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD).

The CWD is the default path to be used for all file operations where a specific path is not provided or as the base when a partial path is used. Each task/batch job maintains its own CWD. Therefore, an initial CWD should be set within the batch file to insure a valid default work destination.

Typing CD or CHDIR without a volume or path displays the CWD (except in a batch file where the output is suppressed).

This command does not support non-NetWare partitions.

CHMOD

(Refer to FLAG documented later in this section.)

CHOWN

List/Set the owner associated with the files matching the search specification. This command utilizes the following format:

CHOWN [[vol:]path\][spec] [user] [opts]

[opts]: /C /DO /FO /H /P /S
(Options must be separated with spaces)

/C Continuous list (default)
/DO Directories only (ignore files)
/FO Files only (ignore directories)
/H Hidden/System files included (default = exclude)
/P Pause listing after each screen (default = /C)
/S Subdirectory recursion

Notes: [spec] can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD). Wild cards supported.

Typing CHOWN without a [user] specification will list the files in the CWD matching [spec] and their associated owners. If [spec] is also excluded, all files (*.*) in the CWD will be listed.

This command does not support non-NetWare partitions.

Entering ABORT CHOWN at the TMConsole (Shell) prompt will terminate any active CHOWN processing.

CLEAR CONNS

Clear (terminate or kill) the specified connection number(s). This command utilizes the following format:

CLEAR CONNS [###] [###] . . .

Notes: If more than one connection number is specified, separate each one with a space.

Output redirection supported.

Examples:

CLEAR CONNS 1 2 3
Clears connection number 1, 2, and 3

CLEAR CONNS 10 26
Clears connection number 10 and 26

CLEAR USERS

Clear (terminate or kill) all active User connections matching the specification. This command utilizes the following format:

CLEAR USERS {user} [user] . . .

Notes: A wild card can be specified (i.e., SUPER* matches SUPERMAN and SUPERVISOR).

If more than one User name is specified, separate each one with a space.

Output redirection supported.

Examples:

CLEAR USERS GUEST
Clears all Users logged in as GUEST

CLEAR CONNS SUPER*
Clears all Users logged in as SUPER, SUPERMAN, SUPERVISOR, etc. - the '*' is a wild card character

COPY

(Refer to XCOPY documentation in this section.)

DEL / DELETE / ERASE

Deletes any files matching the specification. This command utilizes one of the following formats:

DEL[ETE] [[d::]path\]spec [opts]
DEL[ETE] [[vol:]path\]spec [opts]
ERASE [[d:]path\]spec [opts]
ERASE [[vol:]path\]spec [opts]

[opt]: /Y

/Y Confirm intent when prompted for TMConsole (Shell) manual use

Notes: The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

If executed manually at the TMConsole (Shell) prompt and a full wild card (* or *.*) is specified, a prompt will appear requesting confirmation of the operation. The /Y option overrides the prompt, confirming the operation.

Support for the DOS partition requires that the specification includes the drive and path.

DELTREE

Deletes all files and subdirectories matching the specification, including all files and subdirectories which are subordinate to the specified path. This command utilizes the following format:

DELTREE [[vol:]path\][spec] [opts]

[opts]: /# /I /K /P /V /Y
(Options must be separated with spaces)

/# Repeat process up to # times on non-critical error (i.e., NSS can return false results on large subdirectories)
/I Ignore non-critical errors (i.e., continue processing remaining entries after non-critical error)
/K Keep subdirectory structure intact (i.e., only remove files)
/P Purge deleted directories/files
/V Verbose listing (i.e., show processing statistics and expanded error messages)
/Y Confirm intent when prompted for TMConsole (Shell) manual use

Notes: The specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

If [path] is specified without [spec], the entire path is deleted.

If executed manually at the TMConsole (Shell) prompt, a prompt will appear requesting confirmation of the operation. The /Y option overrides the prompt, confirming the operation.

This command does not support non-NetWare partitions.

DIR

Generates a listing of the files matching the search specification. This command utilizes the following format:

DIR [[d:]path\][spec] [opts]
DIR [[vol:]path\][spec] [opts]

[opts]: /A /B /C /D /F /H /L /N /O /P /Q /R /S /T /W /X
(Options must be separated with spaces)

/A Attribute information included in listing
/A= Attribute search (i.e., entries matching Attribute search criteria: /A=? or /A=+? - set, /A=-? - not set)
/B Bare list (name.ext only, use /S to include path)
/C Continuous list (default = /P)
/D Directories included in listing (default, use /-D to exclude - i.e., files only)
/F Files included in listing (default, use /-F to exclude - i.e., directories only)
/H Hidden/System files included (default = exclude)
/L= Length (size) search (i.e., only files equal to specified size or within specified range)
/N Name Space (assigned to file) included in listing
/N= Name Space match used as search criteria
/O Owner information included in listing
/-O Owner invalid (i.e., only entries with invalid or deleted Object ID)
/O= Owner search (i.e., files owner by named Object)
/P Pause listing after each screen (default)
/Q Quiet mode (i.e., suppresses display of individual subdirectory/file entries)
/R Rights information (i.e., only entries with Trustee assignments and their rights)
/-R Rights invalid (i.e., only entries with invalid Trustee assignments)
/R= Rights search (i.e., only entries with Trustee assignments matching specified Object)
/S Subdirectory recursion
/T Time information included in listing
/T= Time search (i.e., only entries with date/time fields matching specification)
/W Wide listing (name.ext only, dirs in brackets)
/X DOS information only (non-DOS information ignored)

Notes: /A will include the attributes information in the listing.

/A= limits the listing to entries matching the specified attribute criteria as follows:
/A=? or /A=+? specified attribute (?) must be set (i.e., /A=h for hidden, /A=RO for Read Only, etc.)
/A=-? specified attribute must be clear (i.e., /A=-h for non-hidden, /A=-RO for non-Read Only, etc.)

Note: Multiple /A= options can be specified with both enabled and disabled attributes concurrently supported.

/L= limits the listing to entries matching the specified file length (size) criteria as follows:
/L=nnnn file length (size) must equal nnnn
/L=+nnnn file length (size) must be equal to or greater than nnnn
/L=-nnnn file length (size) must be equal to or less than nnnn
/L=nnnn-nnnn file length (size) must be within the range of nnnn-nnnn (inclusive).

Note: Only a single /L= option can be specified.

/-O limits the listing to entries with an invalid Owner ID.

/O=”object.cx” limits the listing to entries whose Owner Object ID matching the Object spec (object.cx).

/Q limits the listing to individual subdirectory and directory tree total counts for matching directories and files.

/-R limits the listing to entries with invalid Trustee assignments.

/R=”object.cx” limits the listing to entries with a Trustee Rights assignment matching the Object spec (object.cx).

/T= limits the listing to entries matching the specified date/time criteria as follows:

/T=yyyy/mm/dd	Last Modified date equals yyyy/mm/dd
/T=+yyyy/mm/dd	Last Modified date on or after yyyy/mm/dd
/T=-yyyy/mm/dd	Last Modified date on or before yyyy/mm/dd
/T=yyyy/mm/dd-yyyy/mm/dd	Last Modified date between the specified date ranges (inclusive).
/T=Ayyyy/mm/dd	Last Accessed date equals yyyy/mm/dd
/T=A+yyyy/mm/dd	Last Accessed date on or after yyyy/mm/dd
/T=A-yyyy/mm/dd	Last Accessed date on or before yyyy/mm/dd
/T=Ayyyy/mm/dd-yyyy/mm/dd	Last Accessed date between the specified date ranges (inclusive).
/T=Cyyyy/mm/dd	Created date equals yyyy/mm/dd
/T=C+yyyy/mm/dd	Created date on or after yyyy/mm/dd
/T=C-yyyy/mm/dd	Created date on or before yyyy/mm/dd
/T=Cyyyy/mm/dd-yyyy/mm/dd	Created date between the specified date ranges (inclusive).
/T=Myyyy/mm/dd	Last Modified date equals yyyy/mm/dd
/T=M+yyyy/mm/dd	Last Modified date on or after yyyy/mm/dd
/T=M-yyyy/mm/dd	Last Modified date on or before yyyy/mm/dd
/T=Myyyy/mm/dd-yyyy/mm/dd	Last Modified date between the specified date ranges (inclusive).

Note: Only a single /T= option can be specified.

The Time information (/T) format listing is: Last Modified (Updated), Last Archive, Last Access, and Created.

Multiple options can be combined to limit the listing to a narrow range of entries.

The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

Support for the DOS partition requires that the specification includes the drive and path.

Output redirection supported.

Example:

DIR SYS:*.* /A=h /-D /L=1024-1048576 /O=Admin /T=M2004/06/01-2004/06/30

Recursively (/S) search the SYS: volume for files (/-D excludes directories) with names matching *.* which are Hidden (/A=h) and have a size between 1 KB and 1 MB (/L=1024-1048576) that are owned by Admin (/O-Admin) and were last modified between June 1st, 2004 and June 30th, 2004 (/T=M2004/06/01-2004/06/30). Only files matching all of the criteria specified will be included in the listing.

DUMP

Displays the hex contents of the specified file (similar to DOS DEBUG). This command utilizes the following format:

DUMP [[d:]path\]{spec} [opts]
DUMP [[vol:]path\]{spec} [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)

Notes: The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

Support for the DOS partition requires that the specification includes the drive and path.

ERASE

(Refer to DEL / DELETE documentation in this section)

FILE CLOSE

Closes open files on the Server (NLM or client) which match the file specification. This command utilizes the following format:

FILE CLOSE [[vol:]path\]{spec} [opts]

[opts]: /S

/S Subdirectory recursion

Notes: The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

The file is flushed internally and closed without clearing or terminating any connections.

CAUTION should be exercised when using wild cards since this command will process any file matching the specification, including System and Bindery files.

FILE UNLOCK

Release Server-based locks (NLM or client) for files which match the specification. This command utilizes the following format:

FILE UNLOCK [[vol:]path\]{spec} [opts]

[opts]: /S

/S Subdirectory recursion

Notes: The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

The file is unlocked internally without clearing or terminating any connections.

CAUTION should be exercised when using wild cards since this command will process any file matching the specification, including System and Bindery files.

FIND

Generates a list of files matching the specification which contain the specified string. This command utilizes the following format:

FIND {"string"} [[vol:]path\]{spec} [opts]

[opts]: /C /I /L /N /S /X
(Options must be separated with spaces)

/C Count of lines containing the string
/I Insensitive case compare
/L List matching file names only
/N Number of each line containing the string
/S Subdirectory recursion
/X DOS name display (default = Long name, if it exists)

Notes: The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

The /C option overrides /L and /N, eliminating the display of the matching lines and showing the count with the file name.

This command does not support non-NetWare partitions.

Output redirection supported.

Entering ABORT FIND at the TMConsole (Shell) prompt will terminate any active FIND processing.

FLAG / CHMOD

Changes the attributes associated with the directory entries matching the search specification. This command utilizes the following format:

FLAG [vol:path\[spec]] [[+|-]attributes] [opts]
CHMOD [vol:path\[spec]] [[+|-]attributes] [opts]

[opts]: /C /DO /FO /H /P /S
(Options must be separated with spaces)

/C Continuous list (default = /P)
/DO Directories only (ignore files)
/FO Files only (ignore directories - default)
/H Hidden/System files included (default = exclude)
/P Pause listing after each screen (default)
/S Subdirectory recursion

Notes: The specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

Typing FLAG or CHMOD without any trailing specifications will display a listing of the files in the CWD with their associated attributes.

Typing FLAG or CHMOD with a specification but without any attribute parameters will display a listing of the matching directory entries in the CWD with the associated attributes for the files.

Use the default attribute N (Normal) to reset a file's attributes to Rw only (i.e., FLAG vol:path\spec N).

Up to three sets of attributes enclosed in brackets are displayed (depending upon the entry type and NetWare version). For directories, a single set of attributes is shown. For files, the DOS attributes, NetWare attributes, and Status attributes (for NetWare v4 and later) are shown. The following attributes, if set, are displayed:

Directory attributes:

- Sy - System
- H - Hidden
- P - Purge
- Di - Delete inhibit
- Ri - Rename inhibit
- Dc - Don't compress (NW v4+ only)
- Ic - Immediate compress (NW v4+ only)
- Dm - Don't migrate (NW v4+ only)

(Attributes displayed in the order shown if set)

File Attributes (DOS):

- Ro - Read only
- Rw - Read-write
- H - Hidden
- Sy - System
- A - Archive

File Attributes (NetWare):

- X - eXecute
- T - Transactional
- P - Purge
- Sh - Share
- Di - Delete inhibit
- Ci - Copy inhibit
- Ri - Rename inhibit
- Ra - Read audit (NW v3 only)
- Wa - Write audit (NW v3 only)
- Dc - Don't compress (NW v4+ only)
- Ic - Immediate compress (NW v4+ only)
- Dm - Don't migrate (NW v4+ only)
- Ds - Don't suballocate (NW v4+ only)

File Attributes (NetWare Status):

- Co - Compressed (NW v4+ only)
- Cc - Can't compress (NW v4+ only)
- M - Migrated (NW v4+ only)

(Attributes displayed in the order shown if set)

This command does not support non-NetWare partitions.

Entering ABORT FLAG - or - ABORT CHMOD at the TMConsole (Shell) prompt will terminate any active CHMOD/FLAG processing.

FLAGDIR

Changes the attributes associated with the directories matching the search specification. This command utilizes the following format:

FLAGDIR [[vol:]path] [[+|-]attributes] [opts]

[opts]: /C /H /P /S
(Options must be separated with spaces)

/C Continuous list (default = /P)
/H Hidden/System files included (default = exclude)
/P Pause listing after each screen (default)
/S Subdirectory recursion

Notes: The specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

Typing FLAGDIR without any trailing specifications will display a listing of the directories in the CWD with their associated attributes.

Typing FLAGDIR with a specification but without any attribute parameters will display a listing of the matching directories in the CWD with their associated attributes.

The following attributes, if set, are displayed:

Directory attributes:
Sy - System
H - Hidden
P - Purge
Di - Delete inhibit
Ri - Rename inhibit
Dc - Don't compress (NW v4+ only)
Ic - Immediate compress (NW v4+ only)
Dm - Don't migrate (NW v4+ only)

(Attributes displayed in the order shown if set)

This command does not support non-NetWare partitions.

GRANT

Provides the specified trustee rights to dirs and files matching the search specification for the specified User or Group object. This command utilizes the following format:

GRANT [right] [[vol:]path]{spec} [usr|grp] [opts]

[opts]: None
(Options must be separated with spaces)

Notes: The [[vol:]path] specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD) but must be the second command line parameter. Wild cards supported.

Parameters must be specified in the order shown. The specific trustee rights are listed below:

Trustee rights:

A	- Access
C	- Create
E	- Erase
F	- File Scan
M	- Modify
R	- Read
S	- Supervisor
W	- Write
ALL	- All (of the above rights)

The specified rights are OR'ed with any existing rights the user or group object may already have been granted (i.e., rights are only added, not replaced or removed).

The rights can be specified in any order but must be the first command line parameter and the individual rights must be separated by spaces (i.e., A C E F M, not ACEFM.).

The user or group specification must be a valid Bindery/NDS User or Group object name.

This command does not support non-NetWare partitions.

LIST CONNS

Displays a list of active Server connections matching the connection specification, including connection number, login name, and login time. This command utilizes the following format:

LIST CONNS [spec] [opts]

[opts]: /A /C /O /P
(Options must be separated with spaces)

/A Address (net:node)
/C Continuous list (default = /P)
/O Object type
/P Pause listing after each screen (default)

Notes: If [spec] is provided, it must be a numeric value specifying a single connection to be displayed. The default is to list all of the active connections.

Output redirection supported.

LIST MODULES

Displays a list of loaded Server modules matching the search specification, including name, title, version, compile date, and any additional copyright or description information provided by the developer. This command utilizes the following format:

LIST MODULES [spec] [opts]

[opts]: /C /I /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/I Information about the NLM's allocated resources is included
/P Pause listing after each screen (default)

Notes: If [spec] is provided, it must be a valid module name or search pattern to be used in matching loaded modules (the '*' wild card is supported). The default is to list all of the loaded modules on the Server.

Output redirection supported.

LIST SCREENS

Displays a list of all the module screens on the Server, including which module created the screen and the name of the screen. This command utilizes the following format:

LIST SCREENS [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)

Notes: Output redirection supported.

LIST SESSIONS

Displays a list of all active TaskMaster Remote Console DOS Client sessions on the Server, including the connection/task number, Login name, network:node address, and the Screen being accessed. This command utilizes the following format:

LIST SESSIONS [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/I Information about each active TaskMaster Remote Console DOS Client session
/P Pause listing after each screen (default)

Notes: Output redirection supported.

Not supported by TaskMaster Lite (TMLite).

LIST USERS / USERLIST

Displays a list of User Server connections matching the search specification, including connection number, Login name, and login time. This command utilizes the following format:

LIST USERS [spec] [opts]
USERLIST [spec] [opts]

[opts]: /A /C /O /P
(Options must be separated with spaces)

/A Address (net:node)
/C Continuous list (default = /P)
/O Object type
/P Pause listing after each screen (default)

Notes: If [spec] is provided, it must be a valid User name or search pattern (the '*' wild card is supported). The default is

to list all of the active User connections.

Output redirection supported.

LIST VOLUMES / VOLINFO

Displays a list of the Volumes mounted on the Server matching the search specification, including the number of directory slots in use, used/purgeable/total MB, and Name Space(s) supported. This command utilizes the following format:

LIST VOLUMES [spec] [opts]
VOLINFO [spec] [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)

Notes: If [spec] is provided, it must be a valid Volume name. The default is to list all of the mounted Volumes.

Information about the allocated Directory Slots (entries), Volume Size, In Use Space, Purgeable Space, and Name Space support is provided. (Note: Any Name Space listed in lower case indicates the volume has been configured to support the Name Space but the required Name Space module (.NAM) is not loaded so support is limited.)

Output redirection supported.

MD / MKDIR

Creates the specified directory. This command utilizes the following format:

MD [vol:]{path}
MKDIR [vol:]{path}

[opts]: /D /L /M /N
(Options must be separated with spaces)

/D Create directory as DOS Name Space
/L Create directory as Long (OS2) Name Space (default - if supported by Volume, otherwise default is DOS)
/M Create directory as Macintosh Name Space (if supported by Volume)
/N Create directory as NFS Name Space (if supported by Volume)

Notes: The specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

Support for the DOS partition requires that the specification includes the drive and path.

PURGE

Purges previously erased files within the specified path Current Working Directory (CWD). This command utilizes the following format:

PURGE [[vol:]path\][spec] [opts]

[opts]: /# /A /C /D /P /Q
(Options must be separated with spaces)

/# Purge files deleted more than # days ago
/A All (recursive subdirectory) processing
/C Continuous list (default)
/D Directory name shown only if deleted files purged
/P Pause listing after each screen (default = /C)
/Q Query mode (show deleted file info - none purged)
/Z Zero length files only (i.e., purge empty or NULL deleted files)

Notes: The search specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

A unique search pattern may be specified to limit the processing to specific deleted files. However, it should be noted that the deleted file information is only maintained in DOS Name Space format.

If a search specification is not provided, all previously deleted files are purged either within the specified [[vol:]path\] or the Current Working Directory (CWD), if no [[vol:]path\] specified.

This command does not support non-NetWare partitions.

Output redirection supported.

Entering ABORT PURGE at the TMConsole (Shell) prompt will terminate any active PURGE processing.

RD / RMDIR

Removes the specified directory. This command utilizes one of the following formats:

RD [vol:]{path}
RMDIR [vol:]{path}

Notes: The specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

The operation will fail if any files or subdirectories exist in the directory which is being removed.

Support for the DOS partition requires that the specification includes the drive and path.

REN / RENAME

Renames the source file (the first parameter) to the destination file (the second parameter). This command utilizes one of the following formats:

REN[AME] [[d:]path\]{src} [path\]{dest} [opt]
REN[AME] [[vol:]path\]{src} [path\]{dest} [opt]

[opts]: /C /D /L /M /N
(Options must be separated with spaces)

/C Change owning Name Space as specified
/D DOS Name Space rename (other Name Spaces unchanged)
/L LONG/OS2 Name Space rename (other Name Spaces unchanged)
/M MAC Name Space rename (other Name Spaces unchanged)
/N NFS Name Space rename (other Name Spaces unchanged)
/S Subdirectory recursion

Notes: The source file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards supported.

When using the /C option, another option (/D /L /M /N) must be specified to define the new owning Name Space. Only one Name Space option may be used at a time.

When using the /S option, the specified destination directory must exist as the starting base. Any required subdirectories beneath the destination base will be created as needed to facilitate the renaming of matching files as processing recurses through the source subdirectory tree.

Files cannot be renamed across drives / volumes.

Support for the DOS partition requires that the specification includes the drive and path.

RENDIR

Renames the specified source directory (the first parameter) to the destination directory (the second parameter). This command utilizes the following format:

RENDIR [[vol:]path\]{src} [path\]{dest} [opt]

[opts]: /C /D /L /M /N
(Options must be separated with spaces)

/C Change owning Name Space as specified
/D DOS Name Space rename (others unchanged)
/L LONG/OS2 Name Space rename (others unchanged)
/M MAC Name Space rename (others unchanged)
/N NFS Name Space rename (others unchanged)

Notes: The source file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

When using the /C option, another option (/D /L /M /N) must be specified to define the new owning Name Space. Only one Name Space option may be used at a time.

Directories cannot be renamed across volumes.

This command does not support non-NetWare partitions.

REVOKE

Removes the specified trustee rights from matching directory entries (files or directories) for the specified User or Group object. This command utilizes the following format:

REVOKE [right] [[vol:]path]{spec} [usr|grp] [opts]

[opts]: None
(Options must be separated with spaces)

Notes: The [[vol:]path] specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD) but must be the second command line parameter. Wild cards supported.

Parameters must be specified in the order shown.

The specific trustee rights are listed below:

Trustee rights:

A	- Access
C	- Create
E	- Erase
F	- File Scan
M	- Modify
R	- Read
S	- Supervisor
W	- Write
ALL	- All (of the above rights)

Notes: Specified rights are removed from any existing rights the user or group object may already have been granted (i.e., rights are only removed, not replaced or altered).

The rights can be specified in any order but must be the first command line parameter and the individual rights must be separated by spaces (i.e., A C E F M, not ACEFM.).

The user or group specification must be a valid Bindery/NDS User or Group object name.

This command does not support non-NetWare partitions.

SORT

Rearranges the data records in an ASCII text file into a defined sequence based upon the key patterns provided. This command utilizes the following format:

SORT [[vol:]path\]{src} [[vol:]path\]{dest} {keys}

[opts]: /D /I /N /P
(Options must be separated with spaces)

/D Descending order sort (default = ascending)
/I Case insensitive sort (default = case sensitive)
/N Exclude NULL (blank) records (default = include)
/P Pack data / truncate short keys (default = pad)

Notes: The source and destination file specifications are both required and can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

Up to ten key patterns (keyspec) can be specified to build each key that will be used to rearrange the data records in the file. Key patterns utilize a beginning position and ending position format separated by a hyphen, with each key pattern separated by a comma.

By default, if a key pattern exceeds the record length (i.e., a key pattern of 1-100 is specified but the record is only 80 bytes long), the data returned for the key pattern beyond the End Of Record (bytes 81 through 100) will be padded with spaces (a full 100 bytes are returned). This padding feature can be used to format the key data.

The /P option will force the truncation of shorter data to its actual length (i.e., a key pattern of 1-100 is specified but the record is only 80 bytes long will result in only 80 bytes being returned).

Output redirection supported.

Entering ABORT SORT at the TMConsole (Shell) prompt will terminate any active SORT processing.

SYSTEM CONSOLE

Changes the actively displayed screen on the Server to the System Console screen and, optionally, passes a command line to it for execution. This command utilizes the following format:

SYSTEM CONSOLE [cmdline]

Note: If the actively displayed screen on the Server is not the Server Console screen and is locked (e.g., such as when the screen saver is active), the actively displayed screen is not changed. However, any command line specified is still passed to the System Console for execution.

TASKLIST

Displays the names of currently active and scheduled tasks on the TMConsole screen. This command utilizes the following format:

TASKLIST [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)

Note: For active tasks, the first 60 bytes of any command line options are also displayed.

TMCMD

Launches the trailing Console Command (Local) as a single command task (batch processed). This command utilizes the following format:

TMCMD {command_line}

Note: The trailing {command_line) must be a supported Console Command (Local) with a valid command line.

TMCONFIG

Can be used to change and review the TaskMaster NLM configuration parameters. This command utilizes the following format:

TMCONFIG [opts]

[opts]:	ADD OPERATOR object_name[.cx]	Add TaskMaster Operator to configuration
	DEL OPERATOR object_name[.cx]	Delete TaskMaster Operator from configuration
	LIST OPERATOR [search_spec]	List defined TaskMaster Operators matching the search_spec
	ADD USER object_name[.cx]	Add TaskMaster User to configuration
	DEL USER object_name[.cx]	Delete TaskMaster User from configuration
	LIST USER [search_spec]	List defined TaskMaster Users matching the search_spec
	LOG CLEAR	Clear (empty) the TaskMaster Master log
	LOG SAVE [[vol:]path\]filename.ext	Save TaskMaster Master log as specified then clear (empty) it.

TMREMOTE user_class rights	Define TMRemote access rights for user_class
TMREMOTE LIST	List/show current TMRemote configuration
TMREMOTE PASSWORD	Define TMRemote access password

Notes: By default, using TMSCHEDULE to schedule tasks requires (NDS) Admin or (Bindery) Supervisor, or equivalent, rights. However, the (NDS) Admin or (Bindery) Supervisor, or equivalent user, can define TaskMaster Operators who can then schedule, manage, and execute Tasks.

By default, executing the TaskMaster Remote Console (TMRemote) DOS Client (TMREMOTE.EXE) requires (NDS) Admin or (Bindery) Supervisor, or equivalent, rights. However, the (NDS) Admin or (Bindery) Supervisor, or equivalent, can define TaskMaster Operators and TaskMaster Users who can then use the TMRemote with command line options to execute tasks which reside in one of TaskMaster's Default Search Paths.

(NDS) Admin or (Bindery) Supervisor, or equivalent, rights are required to use the OPERATOR, USER, or TMREMOTE configuration options for this command. Upon execution, a prompt will appear seeking a valid User Name and Password. Once verified and sufficient TaskMaster rights are confirmed, the user can continue to re-execute the command without being re-prompted for User Name and Password until either changing from the TMConsole (Shell) screen or allowing 15 seconds to pass between command executions. In either case, the previous information is cleared and a prompt will re-appear seeking entry of a valid User Name and Password.

Output redirection supported for the LIST options.

Configuring the TMRemote Access Rights:

Using the TMREMOTE option of TMCONFIG, it is possible to define the access rights to the Server Console that are granted to certain users/classes during TMRemote sessions. The specific users/classes are: (NDS) Admin or (Bindery) Supervisor, or equivalent; File Server Console Operators (FSCONSOLE - defined in NDS or Bindery Server properties), and TaskMaster Operators (TMOPERATOR). The access restrictions are based upon the following restrictions (specify the appropriate option for the user/class being configured):

ADMIN | SUPERVISOR:	- FULL (allows unrestricted access) - PASS (allows unrestricted access with password)
FSOPERATOR (Server Console Operator):	- FULL (allows unrestricted access) - PASS (allows unrestricted access with password) - VIEW (allows view only access) - NONE (no access provided)
TMOPERATOR (TaskMaster Operator):	- FULL (allows unrestricted access with password) - VIEW (allows view only access) - PASS (allows view only access with password) - NONE (no access provided)

Examples:

TMCONFIG TMREMOTE ADMIN FULL
Configures TMRemote to grant the Admin / Supervisor (or equivalent) user unrestricted access.

TMCONFIG TMREMOTE FSOPERATOR PASS
Configures TMRemote to allow any user defined as a File Server Console Operator unrestricted access ONLY after entering the correct TMRemote access password.

TMCONFIG TMREMOTE TMOPERATOR VIEW
Configures TMRemote to allow any user defined as a TaskMaster Operator view only access (i.e., the user can change screen and view activity but no keyboard entry is supported).

Configuring the TMRemote Access Password:

Using the TMREMOTE option of TMCONFIG, it is possible to define an access password to further control or restrict TMRemote access rights to the Server Console. To add, change, or delete a password, use the following:

TMCONFIG TMREMOTE PASSWORD

Enter and re-verify the password desired. A null entry in both cases will clear an existing password.

TMHELP

Displays information about the specified TaskMaster Extended Console Command. This command utilizes the following format:

TMHELP [command]

Note: Entering TMHELP alone will display generic help information, including list of the Extended Console Commands.

TMINFO

Displays information about the copy of TaskMaster running on the Local Server.

TMRELOAD

Causes the Server Module to reload (unload/load) itself.

Note: Useful after manually updating the Server Module since the unload / reload process can be accomplished as a single command.

TMRESET

Causes the Server Module to re-read the configuration and schedule files.

Note: This command should only be used if a new schedule or configuration file is copied to the Server, rather than modified through normal means.

TMRUN

Launches Server Module batch processing of the specified Task file. This command utilizes the following format:

TMRUN [[vol:]path\]{task} [arg] [arg] [arg] ...

Notes: If the task specification includes a path (vol:path):
- Search only the specified directory for:
  task; task with the default batch extension (.NCF for NetWare Servers); task with the .TSK extension.

If the task specification does not include a path:
- task must meet Default Search Path Criteria.

Up to ten (10) arguments or parameters ([arg]) can be passed to the task via the command line. Any such [arg] specified on the command line will be placed in the default Dynamic Variables (%0 - %9) in the order specified on the command line.

If a Task uses TMRUN to initiate another Task, both Tasks process concurrently.

This command is not supported by the TMConsole or the System Console if the Secure Server module is loaded.

Hint: If an argument [arg] to be passed to a Task has spaces, enclose [arg] within double quotes. Otherwise, any spaces encountered are considered [arg] separators.

TMSCHEDULE

Review the scheduled task list, delete a scheduled task, or add a new scheduled task. This command utilizes one of the following formats:

```
TMSCHEDULE [spec]
TMSCHEDULE ADD taskname.ext type time
TMSCHEDULE DEL taskspec [/#|/A]

    [opts]:
    ADD   Add the specified task (type and time required):
          type   - date (use ##/## for month/day)
                 - monthly (## for day of each month)
                 - weekly (7 unit table of Y or N entries, SMTWTFS - Sun, Mon, ... Fri, Sat)
          time   - time (use ##:## for hour:minute)
                 - hourly (:## for minutes after each hour)
                 - interval (## for every xx minutes)
    DEL   Delete scheduled tasks matching the task specification (taskspec):
          /#     Delete # (numeric) occurrence of the task
          /A     Delete all matching task occurrences
                 (default is to delete the first occurrence)
```

Notes: Non-DOS Name Spaces are not supported. Scheduled tasks can only be added, deleted, or viewed in the DOS Name Space file name format. Wild cards are supported only when Reviewing scheduled tasks or Deleting scheduled tasks; not when Adding a scheduled task.

The Server Module will only execute task files which reside in the TaskMaster load, Server System, or optional P=/PATH= directories. Therefore, only the Task file name and extension is used (i.e., no Volume:Path information).

Only users previously defined as a TaskMaster Operator or those with Supervisor (Bindery) or Admin (NDS), or equivalent, rights are allowed access to this command.

Examples:

```
TMSCHEDULE
List all tasks in the schedule file (active or not).

TMSCHEDULE WEEK*
List all tasks in the schedule file (active or not) which start with the word WEEK.

TMSCHEDULE ADD NEWYEAR.TSK 01/01 00:01
Schedule the task NEWYEAR.TSK for execution on January 1st at 12:01am.

TMSCHEDULE ADD MONTHLY.TSK 15 :15
Schedule the task MONTHLY.TSK for execution on the 15th of each month at 15 minutes after each hour.

TMSCHEDULE ADD WEEKDAY.TSK NYYYYYN 15
Schedule the task WEEKDAY.TSK for execution on Monday, Tuesday, Wednesday, Thursday, and Friday of each
week at 15 minute intervals.
```

TMSCHEDULE ADD WEEKEND.TSK YNNNNNY 12:00
Schedule the task WEEKEND.TSK for execution on Sunday and Saturday of each week at 12:00pm (noon).

TMSCHEDULE DEL OLDTASK.TSK
Delete the first OLDTASK.TSK entry from the schedule.

TMSCHEDULE DEL OLDTASK.TSK /3
Delete the third OLDTASK.TSK entry from the schedule.

TMSCHEDULE DEL OLDTASK.TSK /A
Delete all of the OLDTASK.TSK entries from the schedule.

TMSCHEDULE DEL OLD*.* /A
Delete all of the entries matching the OLD*.* specification from the schedule.

TMSERVER

Displays and manages configuration and operational information about the Server on which the Server Module is loaded. This command utilizes the following format:

TMSERVER [opt]

[opts]:

CX	Displays the NDS Context for the Server
DS	Displays the full NDS Name for the Server
TREE	Displays the NDS Tree for the Server
UPTIME	Displays the Server Up-Time as a string

Notes: Only the first option specified is processed.

Output redirection supported.

TMSMTP

Submits a simple E-Mail message (text file) to the specified SMTP Server. This command utilizes the following format:

TMSMTP {svc.domain.sub|###.###.###.###} [[vol:]path\]{filename.ext}

Notes: The first specification is the qualified name or dot notated address of the SMTP Server. The second specification is the file specification for the E-Mail message to be sent.

The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

The first lines of the E-Mail message must contain the To: and From: information (i.e., prior to any Date:, Subject:, or E-Mail text). Only one From: (sender) is allowed. Multiple To: (recipient) address lines may be specified. However, only one recipient address per To: line is supported. CC:, BCC:, and Attachments are not supported.

To: and From: addresses may include addressee description strings provided they adhere to standard acceptable formats. At a minimum, the individual E-Mail address must be the last item on the To: or From: address line.

The implementation is of a basic SMTP (Simple Mail Transport Protocol) exchange as defined by the IETF (Internet Engineering Task Force) in RFC 821. No support for user or password authentication is provided.

When used in a task script (batch mode), the ERRORLEVEL condition is set on error.

Not supported by TaskMaster Lite (TMLite).

Examples:

To:/From: (can be used interchangeably in these examples)
Valid: To: “Recipient Description String” <recipientr@recv_domain.com>
Valid: To: “Recipient Description String” recipientr@recv_domain.com
Valid: To: <recipientr@recv_domain.com>
Valid: To: recipientr@recv_domain.com

E-Mail text file (SYS:\SYSTEM\TASKMSTR\EMAIL.TXT):
From: “Sender Description String” <sender@send_domain.com>
To: “Destination Description String” <recipient@recv_domain.com>
Date: Tue, 30 Mar 2004 11:14:34 -0700
Subject: Example E-Mail Message

Here is the text message to be sent...

Note: Date: and Subject: are not required. Multiple To: (recipient) lines can be specified but only one (recipient) address is processed per To: line. The From: (sender) and any To: (recipient) lines must all be specified prior to any Date:, Subject: or E-Mail text data.

Command Line Syntax to send above message
TMSMTP smtp.server.com SYS:\SYSTEM\TASKMSTR\EMAIL.TXT
TMSMTP 192.168.001.123 SYS:\SYSTEM\TASKMSTR\EMAIL.TXT

TMUNLOAD

Causes the Server Module to unload itself.

Note: The only officially supported method for unloading a TaskMaster NLM. By using TMUNLOAD, proper shutdown is insured and the availability of the System Console prompt is not at risk in the event of problems unloading.

TREE

Displays the subdirectory tree structure for the default or specified directory. This command utilizes the following format:

TREE [vol:][path] [opts]

[opts]: /A /B /C /D /F /I /P /S /X
(Options must be separated with spaces)

/A Ascii indent character
/B Bare list (full path listing of each directory)
/C Continuous list (default = /P)
/D Delimited data optional output
/F Fixed length fields optional output (only if /D option specified)
/I Information on directories, files and space used (in B / KB / MB / GB) is included
/P Pause listing after each screen (default)
/S Size not formatted (raw versus B / KB / MB / GB)
/X DOS information only (non-DOS information ignored)

Notes: The path specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). If no path is specified, the CWD is used as the starting point.

Output redirection supported.

TYPE

Displays the contents of the specified file on the TMConsole screen. This command utilizes the following format:

TYPE [[d:]path\]{spec} [opts]
TYPE [[vol:]path\]{spec} [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)

Notes: The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

Support for the DOS partition requires that the file specification includes the drive and path.

USERLIST

(Refer to LIST USERS documentation in this section.)

VOLINFO

(Refer to LIST VOLUMES documentation in this section.)

WHEREIS

Displays all occurrences of file(s) matching the file search specification. This command utilizes the following format:

WHEREIS {spec} [opts]

[opts]: /C /D /F /I /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/D Directory entries included in listing (default, use /-D to exclude - i.e., files only)
/F File entries included in listing (default, use /-F to exclude - i.e., directories only)
/I Info (directory entry) on matching files
/P Pause listing after each screen (default)

Notes: A valid file name or search pattern is required. Wild cards supported.

Output redirection supported.

WHOHAS

Displays which connections (Server module or client) have the specified file open. This command utilizes the following format:

WHOHAS {spec} [opts]

[opts]: /C /P /S
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)
/S Subdirectories recursion will be performed

Notes: A valid file name or search pattern is required. Wild cards supported.

Output redirection supported.

COPY
XCOPY

Copies the source file(s) matching the first parameter to the destination file(s), based upon the second parameter. This command utilizes the following format:

COPY [[d:]path\]{src} [[d:]path\][dest] [opts]
COPY [[vol:]path\]{src} [[vol:]path\][dest] [opts]
XCOPY [[d:]path\]{src} [[d:]path\][dest] [opt]
XCOPY [[vol:]path\]{src} [[vol:]path\][dest] [opt]

[opts]: /D /I /O /T COPY
[opts]: /A /D /E /G /H /I /M /N /O /R /S /T XCOPY
(Options must be separated with spaces)

/A Archive attribute must be set for the source file to be copied
/D Decompress NW Compressed files (default = /-D, leave Compressed)
/E Empty directories will be copied during /S
/G Gridlock destination (i.e., do not automatically close destination for update, fail with InUse if destination open)
/H Hidden/System files included (default = exclude)
/I Ignore errors (multi-file operations, default = abort)
/M Modified files (source Archive attribute set) are copied and the source Archive attribute is cleared
/N Nice mode (open source in shared read mode - default is /-N = exclusive access required)
/O Owner same as source (default. use /-O to skip)
/R Read Only files overwritten (default = error if destination file exists with Read Only attribute)
/S Subdirectories recursion will be performed
/T Trustee Rights same as source (default, use /-T to skip)

Notes: Each specification can be fully qualified (with vol:\path) or relative to the Current Working Directory (CWD). Support for the DOS partition requires that the specification include the drive and path. Wild cards supported.

The following items are synchronized between the source and the destination for any non-DOS partition entry that is copied: NetWare attributes, created date/time, last accessed date/time, last updated date/time, Inherited Rights Mask (IRM), Ownership, and Trustee Rights. (Ownership and Trustee Rights can only be synchronized if both Servers are in the same NDS tree or have corresponding Bindery objects defined. IRM, Ownership, Trustee Rights and all date fields other than last updated date/time are not supported by the DOS-partition.)

Creating new destination subdirectories on the DOS partition with the /E and /S options is only supported on NetWare v4 or later with the modular CLIB loaded.

Output redirection supported.

Entering ABORT COPY | ABORT XCOPY at the TMConsole (Shell) prompt will terminate any active COPY/XCOPY processing.

Examples (XCOPY can always be used in place of COPY but not vice versa):

XCOPY SYS:*.* DATA:*.*
Root of SYS: Volume to root of DATA: Volume

XCOPY SYS:*.LOG C:\SERVER*.*
NW volume to DOS partition

XCOPY C:\SERVER*.* SYS:\SERVER*.*
DOS partition to NW volume

XCOPY *.* \PUBLIC
CWD to \PUBLIC

XCOPY ..*.* *.*
Next directory level higher to the CWD

XCOPY *.BAT *.BAK
Use different extension

XCOPY NEW*.* OLD*.*
Modify the destination name

XCOPY ?123*.* ?456*.*
Modify the destination name

XCOPY C:\
DOS root to CWD

XCOPY . C:\TEMP
CWD to DOS C:\TEMP

XCOPY C:\SERVER BACKUP:\SERVER /E /S
DOS C:\SERVER to NW BACKUP:\SERVER (including all files and directories beneath the source path)

Remote Server Commands

The following Console Command extensions support remote operations with other Servers running a compatible version of the TaskMaster Server Module.

Notes: Server-to-Server operations require that compatible versions of the Server Module must be loaded on both Servers or the operations cannot be performed.

Not supported by TaskMaster Lite (TMLite).

SCOPY
SXCOPY

Copies the file(s) matching the source specification to or from a Remote Server running the Server Module. This command utilizes one of the following formats:

SXCOPY [[vol:]path\]{src} rserver/vol:path\[dest] [opts]
SXCOPY {rserver/vol:path\src} [[vol:]path\][dest] [opts]

[opts]: /B /C /D /I /O /T /+IP[=x] /+NCP[=x] /# SCOPY
[opts]: /A /B /C /D /E /G /H /I /J /M /O /R /S /T /X /+IP[=x] /+NCP[=x] /# SXCOPY
(Options must be separated with spaces)

/A	Archive attribute must be set for the source file to be copied
/B	Block update / delta changes (default, use /-B to disable)
/C	Compress data transfers (-C = disable, default {no option} = intelligent use, /C = all files)
/D	Decompress NetWare compressed files (default = keep compressed, if supported on destination)
/E	Empty directories will be copied during /S
/G	Gridlock destination (i.e., do not automatically close destination for update, fail as InUse if open file)
/H	Hidden/System files included (default = exclude)
/I	Ignore errors (multi-file operations, default = abort)
/J	Jumble (encrypt) file data during transfer
/M	Modified files (source Archive attribute set) are copied and the source Archive attribute is cleared
/N	Nice mode (open source in shared read mode - default is /-N = exclusive access required)
/O	Owner same as source (default. use /-O to skip)
/R	Read Only files overwritten (default = error if destination file exists with Read Only attribute)
/S	Subdirectories recursion will be performed
/T	Trustee Rights same as source (default, use /-T to skip)
/X	eXclude non-DOS name processing (use DOS info only)
/+IP[=x]	IP (if supported by both Servers) optional packet size of # (remove brackets to specify)
/+NCP[=x]	NCPE override (IP support ignored) with optional packet size of # (remove brackets to specify)
/#	Provides processing summary statistics

Notes: The local file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). Wild cards are NOT supported in the source specification when copying from a Remote Server but are supported in the source specification when copying to a Remote Server.

The Block update option (/B) is designed to reduce LAN/WAN exchanges when updating large files are prone to having data appended (i.e., log files, history files, transaction files, etc.). Using pre-read logic and CRC checks, it will compare existing file data to that which is being copied and determine if there are blocks of data which are identical and need not be transferred. The algorithm can significantly improve performance when updating certain files across a slow WAN. The Block update logic is enabled by default and can be disabled using the /-B option.

Compression logic is designed to reduce LAN/WAN data transfers by default, with the options to control its usage:
Default: If no option is specified, each file is compared against known compressed file header patterns. Files which are recognized as already being compressed (i.e., NetWare compressed, .ZIP, .PDF, etc.) are transferred raw (or ‘as is’). An internal, performance optimized compression algorithm is used for the

first few blocks of an uncompressed file and only continued if it yields sufficient compression gains to overcome the minimal overhead required to compress/decompress (i.e., a minimum of 8% required).

/C Compress all data transfers. While even compressed files can typically be compressed further, the overhead required may not justify the savings. Only recommended when the Servers on each end are high performance (P-4 or better) and the link between them is exceptionally slow (under 512 Kbit).

/-C Compression is disabled and all files are transferred raw (or 'as is'). Recommended for slower Servers (P-3 or less) with high performance links (100 Mbit or better).

IP (UDP) is supported for transferring data between properly configured NetWare v5.0 (and later) Servers. The logic will automatically query both Servers (source and destination) to determine the Largest UDP Packet Size supported. Starting with the smaller of the two values as a base from which to test communications, it will attempt to send data to the destination Server, reducing the packet size in steps after each failure, until a successful exchange occurs or the packet size drops below 1 KB, at which point the default NCP Extensions will be used.

Notes: If the source and destination Servers are across a WAN and are configured to support UDP packet sizes larger than the WAN allows, the packet size should automatically reduce to a supported size. To reduce the amount of testing required when the WAN Largest UDP Packet Size is known to be smaller than that supported by the Servers, it is possible to specify the UDP packet size on the command line as follows:

/+IP=? ? = 1 - 32 (max packet size in KBytes) or 1024 - 32768 (max packet size in Bytes)

Since large UDP packets are typically broken into smaller packets of Maximum Transmission Unit (MTU) size by the sending Server then reassembled by the receiving Server (IP fragmentation), some busy WAN links may encounter <Comm> errors with a large UDP packet size, even if supported by the link, due to one or more of the fragmented packets being dropped or lost during the transmission. The communications logic is designed to try to detect and recover from such situations. However, it may prove advantageous to reduce the UDP packet size to minimize the fragmentation of packets, thus reducing the potential for dropped or lost fragments and subsequent <Comm> errors.

If <Comm> errors persist between two Servers and cannot be resolved by adjusting the UDP packet size, it is possible to force the logic to use NCP Extensions (NCPE) which provides guaranteed delivery using the NCP Protocol, although much slower than possible via IP (UDP). To force the logic to use the NCPE delivery method, use the following command line option:

/+NCP=? ? = 1 - 32 (max packet size in KBytes) or 1024 - 32768 (max packet size in Bytes)

The packet size should not need to be specified unless <Comm> errors persist over slow WAN links.

The following items are synchronized between the source and the destination for any entry that is copied: NetWare attributes, created date/time, last accessed date/time, last updated date/time, Inherited Rights Mask (IRM), Ownership, and Trustee Rights. (Ownership and Trustee Rights can only be synchronized if both Servers are in the same NDS tree or have corresponding Bindery objects defined.)

If an error occurs during the copy process, a message will be displayed indicating whether a Read (source) or Write (destination) error occurred and the name of the file. By default, processing terminates and the ERRORLEVEL flag is set if an error is encountered.

A file cannot be copied between two Remote Servers by a third Server (unless the TMSCMD command is used to submit the appropriate command to the source Server).

The following options are not supported when copying files from a Remote Server to the Local Server:

/A /E /M /S

These options are only supported when copying files from the Local Server to a Remote Server.

This command does not support non-NetWare partitions.

Output redirection supported.

Entering ABORT SCOPY or ABORT SXCOPY at the TMConsole (Shell) prompt will terminate any active SCOPY/SXCOPY processing.

Example - Copy file(s) to Remote Server (RSERVER - either SCOPY or SXCOPY can be used):

```
SxCOPY vol:path\file.ext RSERVER/vol:path\file.ext
SxCOPY vol:path\file.ext RSERVER/vol:path
SxCOPY vol:path\*.dat RSERVER/vol:path\*.bak
SxCOPY vol:path\*.* RSERVER/vol:path
SxCOPY vol:path RSERVER/vol:path
```

Example - Copy file from Remote Server (RSERVER - either SCOPY or SXCOPY can be used):

```
SxCOPY RSERVER/vol:path\file.ext vol:path\file.ext
SxCOPY RSERVER/vol:path\file.ext vol:path
```

STASKLIST

Displays on the TMConsole screen the names of currently active and scheduled TaskMaster tasks from the specified Remote Server. This command utilizes the following format:

STASKLIST {rserver} [opts]

[opts]: /C /P
(Options must be separated with spaces)

/C Continuous list (default = /P)
/P Pause listing after each screen (default)

Notes: Remote Server must be specified (rserver) and must be running a compatible version of the Server Module.

STMINFO

Displays information about the copy of TaskMaster running on the specified Remote Server. This command utilizes the following format:

STMINFO {rserver}

Notes: Remote Server must be specified (rserver) and must be running a compatible version of the Server Module.

TMSCMD

Execute the specified command line on the designated Remote Server. This command utilizes the following format:

TMSCMD {rserver} {command}

Notes: The command should be exactly the same as if entered at the Remote Server's TMConsole or System Console.

Remote Server must be specified (rserver) and must be running a compatible version of the Server Module.

Executing .TSK scripts on the Remote Server requires the TMRUN command preface, just as if it were entered on the Remote Server via the TMConsole or the System Console.

The ERRORLEVEL condition is set based upon whether the command could be executed.

Examples:

TMSCMD SERVER_312 load monitor
Send the command 'load monitor' to SERVER_312.

TMSCMD SERVER_410 tmrun dbbackup.tsk
Send the command 'tmrun dbbackup.tsk' to SERVER_410.

TMUPDATE

Simplifies the distribution of Server Module updates. The Server Module and Help files associated with the currently loaded TaskMaster NLM are copied to Remote Servers then the TMRELOAD command is submitted to the Remote Server to force a reload of the Server Module, effectively loading the new update. This command utilizes the following format:

TMUPDATE [rserver] [rserver] [rserver] ... [opts]

[opts]: /U
(Options must be separated with spaces)

/U Update / Upgrade only (i.e., skip Servers running same/newer release, default = update all matching Servers)

Notes: Multiple Server names can be specified on the command line, if separated by spaces. Wild cards are supported(i.e., AVANTI*). If no Server specification is provided, all known Servers are checked.

If a mix of Master and Secure Server Modules are in use on the network, both Server Module files should be available in the load directory. The TMUPDATE command will identify which Server Module (Master or Secure) is loaded on the Remote Server and attempt to distribute the appropriate file.

Output redirection supported.

Chapter 5 - File Replication / Data Synchronization

SYNC is actually just one of the TMConsole (Shell) extended Console Commands found in the Full version of TaskMaster. However, its functionality and popularity has propelled it to stature as a specifically identified feature in TaskMaster.

Note: SYNC is not supported by TaskMaster Lite (TMLite).

SYNC

SYNC can be used to synchronize directory contents and file data in two separate directories. The directories can be on the same Volume, across Volumes of the same Server, or across Servers on the network, even across NDS Trees. This command utilizes the following format:

```
SYNC  [[vol:]path\]{spec}  {vol:path}  [opts]  [<excl.txt]
SYNC  [[vol:]path\]{spec}  [rserver/]{vol:path}  [opts]  [<excl.txt]
SYNC LIST
```

[opts]: /A /B /C /D /E /F /H /I /J /K /L /M /O /Q /R /S /T /U /V /W /X
/+DSR=[Y|N] /+ENT=x /+EXC=x /+IP[=x] /+MIG=[Y|N] /+NCP[=x]
(Options must be separated with spaces)

/A	Add source entries not existing on destination
/B	Block level (delta) change update logic (default, /-B = full copy)
/C	Compress data transfers (-C = disable, default {no option} = intelligent use, /C = all files)
/D	Delete destination entries not existing on source
/E	Exclusive source access required (default, use /-E for shared ReadOnly access)
/F	File data comparison (compare date/time & size only, ignore attributes)
/G	Gridlock destination file (i.e., do not automatically close destination if open, fail as InUse)
/H	Hidden/System entries included (default = exclude)
/I	Ignore errors (proceed with next entry, default = abort)
/J	Jumble file data (encrypt data during transfer)
/K	Keep subdirs (use with /D to exclude subdir deletion)
/L	Log SYNC command line in output redirection file
/M	Modified source entries are copied to destination
/O	Owner = source, if same tree (default, /-O = skip)
/Q	Query mode (only show processing needed)
/R	Replace destination with matching source files
/S	Subdirectories recursion will be performed
/T	Trustee rights = source, if same tree (default, /-T = skip)
/U	Update the source with newer destination entries
/V	Verify files (update destination if different)
/W	Warnings/Errors only (i.e., limit output to warnings/errors)
/X	eXclude non-DOS name processing (use DOS info only)
/+DSR=[Y|N]	replicate Directory Space Restrictions (Directory Quotas) from source to destination
/+ENT=x	maximum ENTries allowed in a single subdirectory within a directory tree (default = 32768)
/+EXC=x	maximum number of resolved entries allowed in the EXClusion list (default = 16384)
/+IP[=x]	IP preferred (if supported by both Servers) with optional packet size of # (exclude brackets)
/+MIG=[Y|N]	MIGrated files are processed (default, /+MIG=Y) or ignored (/+MIG=N) (exclude brackets)
/+NCP[=x]	NCP Extension override (IP support ignored) with optional packet size of # (exclude brackets)
LIST	LIST (display) summary status for all active SYNC processes

Notes: The source directory MUST reside on the Local Server and MUST be the first specification. The destination can be another directory on the same Volume, on a different Volume on the Local Server, or on a Remote Server. If the destination is on a Remote Server, the Server name must be included along with the Volume and Path in the

destination specification. Wild cards are supported for the source specification, but not the destination. Any wild cards specified on the source will be used for the destination matches.

Each file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD). The destination specification must be a valid Server/Volume:Path or Volume:Path combination.

There are no default processing options. If no processing options are specified, processing does not occur. Unless /Q is specified, any entries matching the specified option criteria will be processed with the following exceptions: NetWare Server or Volume specific system files, such as SYS:_SWAP_.MEM, SYS:VOLDATA.TDF, SYS:VOL$LOG.ERR, SYS:BACKOUT.TTS, and other common system files, are automatically excluded from SYNC processing.

The Block update option (/B) is designed to reduce LAN/WAN exchanges when updating large files that are prone to having data appended (i.e., log files, history files, transaction files, etc.). Using pre-read logic and CRC checks, it will compare existing file data to that which is being copied and determine if there are blocks of data which are identical and need not be transferred. The algorithm can significantly improve performance when updating certain files across a slow WAN. The Block update logic is enabled by default and can be disabled using the /-B option.

Compression logic is designed to reduce LAN/WAN data transfers by default, with the options to control its usage:

Default: If no option is specified, each file is compared against known compressed file header patterns. Files which are recognized as already being compressed (i.e., NetWare compressed, .ZIP, .PDF, etc.) are transferred raw (or 'as is'). An internal, performance optimized compression algorithm is used for the first few blocks of an uncompressed file and only continued if it yields sufficient compression gains to overcome the minimal overhead required to compress/decompress (i.e., a minimum of 8% required).

/C Compress all data transfers. While even compressed files can typically be compressed further, the overhead required may not justify the savings. Only recommended when the Servers are not on the same Local Area Network (LAN) and are both running high performance CPUs.

/-C Compression is disabled and all files are transferred raw (or 'as is'). Recommended for high performance links (100 Mbit or better).

The Exclusive source file access required option (/E) is enabled by default. When enabled, this option requires that SYNC be able to gain exclusive access to the source file before it can copy the contents otherwise the file is tagged as being <InUse> and is not processed. This is the safest, most secure method of insuring data integrity. Disabling this default method of operation via the /-E option allows SYNC to access source files using a shared Read Only method which provides access to many files normally bypassed as being <InUse>. However, caution should be exercised when using this option during heavy periods of user activity with files that are frequently modified as a file being copied by SYNC could possibly be updated by another process or user during the operation, potentially compromising the destination file contents. It is best utilized only during off hours when the chances are less likely that any files accessed in this shared manner might be modified while being processed.

Note: NetWare Compressed files have to be accessed in exclusive mode to maintain compression.

IP (UDP) is supported for transferring data between properly configured NetWare v5.0 (and later) Servers. The logic will automatically query both Servers (source and destination) to determine the Largest UDP Packet Size supported. Starting with the smaller of the two values as a base from which to test communications, it will attempt to send data to the destination Server, reducing the packet size in steps after each failure, until a successful exchange occurs or the packet size drops below 1 KB, at which point the default NCP Extensions will be used.

Notes: If the source and destination Servers are across a WAN and are configured to support UDP packet sizes larger than the WAN allows, the packet size should automatically reduce to a supported size. To reduce the amount of testing required when the WAN Largest UDP Packet Size is known to be smaller than that supported by the Servers, it is possible to specify the UDP packet size on the command line as follows:

/+IP=? ? = 1 - 32 (max packet size in KBytes) or 1024 - 32768 (max packet size in Bytes)

Since large UDP packets are typically broken into smaller packets of Maximum Transmission Unit (MTU) size by the sending Server then reassembled by the receiving Server (IP fragmentation), some

busy WAN links may encounter <Comm> errors with a large UDP packet size, even if supported by the link, due to one or more of the fragmented packets being dropped or lost during the transmission. The communications logic is designed to try to detect and recover from such situations. However, it may prove advantageous to reduce the UDP packet size to minimize the fragmentation of packets, thus reducing the potential for dropped or lost fragments and subsequent <Comm> errors.

If <Comm> errors persist between two Servers and cannot be resolved by adjusting the UDP packet size, it is possible to force the logic to use NCP Extensions (NCPE) which provides guaranteed delivery using the NCP Protocol, although much slower than possible via IP (UDP). To force the logic to use the NCPE delivery method, use the following command line option:

/+NCP=? ? = 1 - 32 (max packet size in KBytes) or 1024 - 32768 (max packet size in Bytes)

The packet size should not need to be specified unless <Comm> errors persist over slow WAN links.

Support for processing of migrated files (files migrated to external storage - NetWare Extended Attribute M set) is enabled (/+MIG=Y) by default (specifying the option is not required). Disabling the support for processing of migrated files is optional (to ignore migrated files, use the /+MIG=N option).

The order the options are specified does not affect processing or alleviate option conflicts. Conflicting options are processed in the following order:

- Replacement (/R): Destination matching files are replaced regardless of any other options specified.
- Verification (/V): Date/time, or size difference in the destination and source.
- Modification (/M): Date/time stamp is newer on the source than on the destination.
- Update (/U): Date/time stamp is newer on the destination than on the source.

As a result, conditions for the Update (/U) option will never be met when the Replace (/R) or Verify (/V) options are used since replacement will occur prior to the update test. In some situations, the Modification (/M) option may also hinder Update (/U) processing.

The following items are synchronized between the source and the destination for any entry that is copied: Standard file attributes, NetWare Extended Attributes (EA), created date/time, last accessed date/time, last updated date/time, Directory Space Restrictions (a.k.a., Directory Quotas), Inherited Rights Mask (IRM), Ownership, and Trustee Rights. (Ownership and Trustee Rights can only be synchronized if both Servers are in the same NDS tree or have corresponding Bindery objects defined.)

Using input redirection (<vol:path\exclusion.lst), specific files, file patterns, and directories can be excluded from SYNC processing. The exclusion list must be an ASCII text file containing one specification per line which matches the source file(s) to be excluded from the SYNC process. Wild cards are supported in the exclusion specification. The following criteria or rules apply:

- Exclusion entries are specific to the source. Destination entries cannot be excluded.
- NetWare Server or Volume specific system files, such as SYS:_SWAP_.MEM, SYS:VOLDATA.TDF, SYS:VOL$LOG.ERR, SYS:BACKOUT.TTS, and other common system files, are automatically excluded.
- A generic file pattern (*.JPG) excludes any files in the base source directory which match the pattern. If the /S option is appended to the file pattern (i.e., *.JPG /S), any files matching the file pattern which reside in the subdirectory tree beneath the source are also excluded.
- A path specific file or file pattern (e.g., SYS:SYSTEM\DS.NLM or SYS:SYSTEM\DS*.NLM) excludes any files in the specified directory which match (i.e., both examples will exclude the SYS:SYSTEM\DS.NLM file). If the /S option is appended to the specification (i.e., SYS:SYSTEM\DS.NLM /S), any files matching the file pattern which reside in the subdirectory tree beneath the specified path (SYS:SYSTEM) are also excluded.
- A subdirectory specification without wild cards or file pattern (e.g., VOL:PATH) will exclude all directories and files in the specified subdirectory, including any which reside beneath the specification. It is more efficient to specify only the subdirectory instead of the subdirectory with a generic all file wild card (i.e., VOL:PATH is more efficient than VOL:PATH*.*) as both provide the same exclusion results but the wild card specification requires additional processing.

Input redirection of an exclusion file must follow all other options (except output redirection). A valid file specification must be provided (no wild cards). The file specification can be either fully qualified (vol:path) or relative to the Current Working Directory (CWD).

Output redirection supported. Each SYNC process is allocated an NLM screen for console display whether output redirection is specified or not. When output redirection is specified, the processing activity normally displayed on this screen is redirected to the specified log file and a real-time status message is displayed. To view processing activity when output redirection is specified, change to the desired SYNC process specific screen and press the INS key. To disable the activity display, press the DEL key. Note: These key options are only available when output redirection is specified and are specific to the SYNC process associated with the NLM screen on which the keys were pressed. Output redirection is not affected by the use of the INS/DEL keys.

This command does not support non-NetWare partitions.

Entering ABORT [taskname] to terminate an active task that has a SYNC in process will automatically terminate the SYNC process prior to terminating the task. A separate ABORT SYNC is no longer required.

Entering ABORT SYNC at the TMConsole (Shell) prompt will terminate any active SYNC processing. To ABORT a specific SYNC process when multiple, concurrent SYNC processes are active, first use the SYNC LIST command to identify the SYNC process to be terminated then use ABORT SYNC /# where # is the number of the SYNC process (as displayed by SYNC LIST) to be terminated.

For more in depth information and examples on using the SYNC Console Command, review the SYNC FAQ (URL link available on the Support page on the WEB Site).

Examples:

SYNC SYS:APP*.* BACKUP:APP /A
Add any files which exist solely on the source directory (SYS:APP) to the destination directory (BACKUP:APP).

SYNC SYS:APP*.* BACKUP:APP /M
Modified files (exist in both directories but the file on SYS:APP is newer) are copied.

SYNC SYS:APP*.* BACKUP:APP /R
Replace all files which exist in both directories, copying from the source (SYS:APP) to the destination (BACKUP:APP).

SYNC SYS:APP*.* FS2/SYS:APP /A /D /M /S
Synchronize the local SYS:APP and Remote Server FS2/SYS:APP directories as follows: Add any files from SYS:APP which do not exist on FS2/SYS:APP; Delete any files on FS2/SYS:APP which do not reside in SYS:APP; Modified files from SYS:APP are to be copied over their counterparts in FS2/SYS:APP; And, Subdirectory processing (traversing the entire directory tree below SYS:APP) is enabled.

SYNC SYS:APP*.* FS2/SYS:APP /A /D /M /S <SYS:SYNC.EXC
Same as above except that any files or directories matching the specifications in the exclusion list (SYS:SYNC.EXC input redirection file) are not processed. The exclusion list is built from the files in the source directory which match the specification. Since the Delete option (/D) is specified in this example, files which exist on the destination but not the source will be deleted. Therefore, using an exclusion specification with a wild card will only protect files that exist on both the source and destination. To protect destination files which do not exist on the source from being deleted in a directory, specify the directory in the exclusion list (without any file specification or wild cards).

SYNC APP:*.* FS2/APP: /A /D /H /I /L /S /V <SYS:SYNC.EXC >SYS:SYNC.LOG
Synchronize the local APP: and Remote Server FS2/APP: volumes as follows: Add (/A) any files from APP: which do not exist on FS2/APP:; Delete (/D) any files on FS2/APP: which do not reside on APP:; Hidden and System (/H) files are to be processed; Ignore (/I) any non-critical errors (i.e., File In Use); Log (/L) the command line in the output redirection file (SYS:SYNC.LOG); Traverse (recurse) the Subdirectory (/S) tree; Verify (/V) files replacing any destination files which are not identical to the source; And redirect all of the output normally shown on the Console to the SYS:SYNC.LOG file.
(Note: This is the most common usage providing essentially a mirror copy with full logging of the processing.)

Chapter 6 - Task Management Windows Client

The **TaskMaster**® Task Management Windows Client (TMClient.EXE) provides the means to monitor, schedule and terminate Tasks via a workstation. Both NetWare .NCF and extended TaskMaster .TSK Tasks can be scheduled for execution on a specific date, day of the month, or specific days of the week, and at a particular time, at a certain minute after each hour, or at intervals (every xx minutes), as well as monitored and terminated.

When executed, TMClient displays a dialog box with a Server selection box and two list boxes: The Active Tasks list box shows the status of actively running Tasks while the Scheduled Tasks list box shows currently scheduled Tasks.

Server Selection

Click on the enclosed box titled “Select a TaskMaster Server” to select an initial Server. A list box of Server connections which have a compatible version of the TaskMaster NLM loaded will appear. Position the mouse pointer over a Server entry and press the left mouse button to select the Server or press the right mouse button to view the version of NetWare and TaskMaster installed on the Server. The Server name will appear in the enclosed box. To change between Servers, click on the enclosed box with the current Server name displayed and follow the previous instructions.

Active Tasks List Box

The Active Tasks list box shows currently running Tasks which are being processed by TaskMaster on the selected Server and is automatically updated every second. The Task name, the elapsed time running, the current line number / command being processed, and any command line options specified when the task was executed, as well as the full path for the running task, are shown.

Positioning the mouse pointer over an Active Task entry and pressing the left mouse button one time highlights (selects) the entry. Positioning the mouse pointer over an Active Task entry and clicking the right mouse button one time displays a message box containing the full path name (vol:path\filename.ext) of the Active Task.

Aborting An Active Task
Either clicking the left mouse button when the mouse pointer is positioned over the Abort button after highlighting an Active Task entry or positioning the mouse pointer over an Active Task entry and double clicking the left mouse button will prompt for confirmation that the selected task should be terminated. Confirming the decision will set the Abort flag for the Task.

Note: Task termination may not occur immediately. In many cases, the currently executing batch command must complete before the TaskMaster processor will detect that the Abort flag has been set and terminate the Task. If the Task was manually executed via the TMRUN command, the Tasks NLM Screen must either time out or be closed manually before the Task will terminate.

Scheduled Tasks List Box

The Scheduled Tasks list box shows the Tasks currently scheduled for TaskMaster to process on the selected Server. The Task name and schedule information are shown. The Scheduled Task list box is only updated when a change is made to the schedule or the mouse pointer is positioned over the Refresh button and the left mouse button is clicked.

Add a Scheduled Task
After selecting the appropriate Server to review the Scheduled Tasks, position the mouse pointer over the Add button and click the left mouse button once to activate the Add Scheduled Task dialog box. Select a TaskMaster NLM supported directory for scheduled tasks from the pull down list. Upon selecting a directory, a file dialog box listing all the .TSK Files (*.tsk) residing in the directory will appear. The search criteria for the selected directory can be altered to .NCF Files (*.ncf) or All Files (*.*) by changing the default selection in the ‘Files of type’ pull down list.

Browse from among the displayed files to locate the desired Task then either position the mouse pointer over the desired

file and double click the left mouse button to highlight and accept the selection; or click the left mouse button once to highlight the file then move the mouse pointer over the Open button and double click the left mouse button to accept the highlighted file. The selected file will be displayed underneath the TaskMaster NLM supported directory pull down list.

Enter the appropriate date and time for the scheduled task, noting the examples provided, then position the mouse pointer over the OK button and click the left mouse button to accept the specified schedule. If an error was made in the schedule syntax, an appropriate message box will appear and control is returned to the date/time entry box. Otherwise, the task schedule and the Scheduled Task list box are updated accordingly.

Copy a Scheduled Task

After selecting the appropriate Server to review the Scheduled Tasks, position the mouse pointer over the Task to be copied then click the left mouse button once to highlight the selection. Re-position the mouse pointer over the Copy button then click the left mouse button. A list of attached Servers with the TaskMaster NLM loaded will appear. Select one of the Servers by positioning the mouse pointer over the Server name then clicking the left mouse button once.

- If a different Server is selected, the scheduled information for the selected task will be copied to the specified Server. Note: The task file is not copied, only the schedule information for the task. Remember to verify that a copy of the task file exists on the Server where the task was scheduled.

- If the same Server is selected, a prompt will appear confirming duplication of the already scheduled task. Upon confirmation, a second prompt will appear providing the opportunity to edit the duplicated task schedule. The Scheduled Tasks list box will then be updated to reflect the added scheduled task.
  Note: Rare is the need to schedule a single task for multiple concurrent execution so edit the schedule accordingly.

Delete a Scheduled Task

After selecting the appropriate Server to review the Scheduled Tasks, position the mouse pointer over the Task to be deleted then click the left mouse button once to highlight the selection. Re-position the mouse pointer over the Delete button then click the left mouse button. A prompt will appear requiring confirmation of the deletion. The Scheduled Task list box will then be updated accordingly.

Edit a Scheduled Task

After selecting the appropriate Server to review the Scheduled Tasks, position the mouse pointer over the Task to be edited and either double click the left mouse button or click the left mouse button once to highlight the selection, then position the mouse pointer over the Edit button and click the left mouse button.

Enter the appropriate date and time for the scheduled task, noting the examples provided, then position the mouse pointer over the OK button and click the left mouse button to accept the specified schedule. If an error was made in the schedule syntax, an appropriate message box will appear and control is returned to the date/time entry box. Otherwise, the task schedule and the Scheduled Task list box are updated accordingly.

SYNC Jobs List Box

The SYNC Jobs list box shows currently running SYNC Jobs which are being processed by TaskMaster on the selected Server and is automatically updated every second. The Source and Destination from the command line, the elapsed time, the directories examined / processed, the files examined / processed, the amount of file data transferred, and the number of errors encountered, as well as the command line options specified for the SYNC Job, are shown.

Chapter 7 - Remote Console DOS Client

The **TaskMaster**® Remote Console DOS Client (TMRemote.EXE) provides a more flexible, functional, secure, and lower resource alternative to access the Server Console screens from a workstation than available through Novell's RCONSOLE utility. It also offers the ability to submit commands or tasks directly to the Server in either a manual or batch mode. It has the following features:

- Allows multiple, simultaneous sessions under Windows (to single or multiple Servers running a Server Module)
- Provides multiple levels of Bindery/NDS aware security, including View Only modes of operation (full view of the Server Console screens but no keyin entry accepted from the workstation)
- Can maintain synchronization with the active screen on the Server Console or be locked to a specific NLM screen
- Supports command line options for batch submission of Server Console commands or task execution (various levels of security also regulate these options)

Note: Not supported by TaskMaster Lite (TMLite).

Starting the Remote Console DOS Client

The Remote Console DOS Client can be launched in the same fashion as any DOS compatible program.

Remote Console DOS Command Line
To start the Remote Console DOS Client from the DOS command line, follow these steps:

Enter the Remote Console DOS Client's Full Path
Type the Remote Console DOS Client's drive and directory, followed by "TMREMOTE.EXE". If the default setup suggestion was used, type SYS:\SYSTEM\TASKMSTR\TMREMOTE.EXE and press <Enter>. When the program is started, it will attempt to establish a remote session with the default Server.

Command Line Options

The following command-line options are supported:

TMREMOTE [opts]

[opts]: S={server} P={password} C="command string" T=taskname[.ext] ### /V -?
(Options must be separated with spaces)

S= Server with which to start Remote Console session (default = Primary Server, if not Server mapped drive)
P= Password for Remote Console access (if configured, prompt will appear if configured and not specified)
C= Batch: Command to send to TMConsole (Shell) screen
Note: User must meet the criteria for unrestricted access rights. If the command includes multiple parameters or has any embedded spaces, enclose the entire string in double quotes (“”).
T= Batch: Task to submit for processing
Note: User must be Supervisor (Bindery), Admin (NDS), or equivalent, or have been previously defined as a TaskMaster Operator or TaskMaster User. Task file name only (DOS 8.3 name). Task file must reside in a supported search directory.
Refresh rate for screen updates (millisecs: default = 200, min = 100, max = 60000)
/V Batch: Start Remote Console session after sending batch request
-? Help (activated by: ?, -?, help, or -help)

Notes: If a Server is not specified on the command line, a list box of active Server connections will appear. Select a Server from this list or press the Insert key to bring up a list box of other known Servers which can be logged into from within the utility.

To fully automate batch processing (C= or T=), a Server must be specified on the command line. By default, batch processing will terminate immediately after submitting the batch request without activating a remote session. If the /V (/VIEW) option is specified, a remote session will be activated (normal access controls in place). Note: To submit a command or task to the Server via the TMRemote command line required Admin / Supervisor (or equivalent) rights or the user must have been previously defined as a TaskMaster Operator (TMOPERATOR) or TaskMaster User (TMUSER).

The frequency at which the Server is polled does not usually need to be adjusted. However, it can be slowed down by increasing or sped up by decreasing the default frequency of 200ms (the higher the value the less frequently the Server will be polled). This can be useful if the utility is only being used to monitor the Server Console on a period (non-real-time) basis.

The utility displays the active screen on the Server Console. Therefore, if someone changes the active screen on the Server Console, the utility will reflect that change. To retain the selected screen in such situations, enable the keyboard Scroll Lock. If Scroll Lock is active, the selected screen is retained regardless of what happens at the Server Console.

The utility respects NetWare Server Console security and will not change from a locked screen until it is unlocked. Pressing Alt-F1 only show the active screen, if locked.

IMPORTANT: The Remote Console DOS Client is a character based utility. It does not support interaction with NetWare v5/v6 GUI screens in this release. While logic has been implemented to avoid conflicts, caution should be exercised against attempting access to NetWare v5/v6 GUI screens since the results may be indeterminate.

Keyboard Commands

The Remote Console DOS Client supports keyboard commands similar to that supported at the Server Console and in Novell's RCONSOLE utility, including:

Commands	Description
<Alt> + <F1>	Activate list box of active screens
<Alt> + <F2>	Terminate the active remote session
<Ctrl> + <End>	Terminate the active remote session
<Alt> + <F3>	Scroll forward one NLM screen
<Alt> + <+>	Scroll forward one NLM screen
<Alt> + <F4>	Scroll backward one NLM screen
<Alt> + <->	Scroll backward one NLM screen
<Alt>	Display active NLM screen name
<Scroll Lock>	Continue to display the selected screen even if the active screen on the Server Console changes

Configuring Access Password

Using the TMCONFIG command, the Supervisor (Bindery) or Admin (NDS), or equivalent, user can define a TMRemote access password to further control or restrict access rights to the Server Console. (Review the Console Commands chapter, Local Server Commands - TMCONFIG section for more information on configuring the TMRemote access password.)

Configuring Access Rights

Using the TMCONFIG command, the Supervisor (Bindery) or Admin (NDS), or equivalent, user can define what access rights to the Server Console are granted to users during TMRemote sessions based upon a number of criteria. (Review the Console Commands chapter, Local Server Commands - TMCONFIG section for more information on configuring the TMRemote access rights.)

Chapter 8 - Support

Several checks have been incorporated into the installation procedure, Client programs, and the Server Modules to insure compatibility with the installation environment. Most problems encountered during the installation and use of this product are a result of these checks limiting the operation of the software.

Troubleshoot Installation

The user must be logged in to the Server to which the software is to be installed as SUPERVISOR (Bindery), Admin (eDirectory / NDS), or a User with equivalent rights. This insures that the files can be copied to the destination locations.

The workstation must have a drive mapped to the Server. The installation utility uses Windows Client Copy logic which requires that a physical or logical drive be mapped to the file copy destination.

Troubleshoot Server Module Operation

The NetWare Server Module should load, either manually or via AUTOEXEC.NCF, without conflict on supported Server platforms. The requirements for the NLM are as follows:

- NetWare v6.5/OES: Latest released Support Pack and Post-SP NSS patches recommended

Troubleshoot Remote Console DOS Client Operation

To support the NetWare environment, the workstation must have the latest Novell NetWare Client loaded and be logged into the desired Server. The Remote Console DOS Client will attempt to establish a proxy session with the default Server.

The default NetWare Server is defined as follows:

- If the current drive is a network drive, it is the Server on which the network drive is mapped (Resource Server).
- If the current drive is not a network drive, it is the Server where the user initially logged in (Primary Server).

The workstation must be attached to a Server with the Server Module loaded before a remote session can be established.

Other error messages which may be encountered:

- NWCalls initialization failure (0xhhhh)!

  Cause: Novell NetWare Client is not properly installed/loaded on the workstation.
  Client not longed into NetWare Server as Primary Login connection.

  Note: If the error persists once the client driver has been loaded, note the error code and contact technical support.

- Incompatible Client/Server program versions

  Cause: There is a version discrepancy between the Server Module loaded on the Server and the Remote Console DOS Client (both versions must match). Reinstall the software and reload the Server Module.

  Note: If you have recently upgraded/updated the software, it is likely that the previous Server Module needs to be unloaded and the newer Server Module loaded.

Contact Information

Product support is available to registered Users. To insure eligibility, Users **MUST** register the software by completing and returning the Product Registration Card within the first thirty days following receipt of the software. Failure to comply may result in the refusal of technical support until this requirement has been fulfilled.

Every effort is made to thoroughly test every product prior to shipment. However, problems may occur in spite of the best quality assurance. In addition, the integration of new features and the evolution of the hardware / software which it supports can also affect the product.

The technical support staff endeavors to respond quickly to any inquiries on a First In, First Out (FIFO) basis. Having the following information readily available, prior to initiating technical support contact, will help produce a quicker response to support queries:

- NetWare Server: NetWare version / Support Pack
- TaskMaster: Module version / Serial Number

If the problem is TaskMaster Client related, please also provide:

- Workstation: Workstation OS version / NetWare Client version
- TaskMaster: Client version

Also, the more in-depth the problem description, including any error message, the quicker the chances are for resolution. Technical support is available by eMail, fax, and phone (the latter for users within the initial 90 days following installation and subscribers to Annual Software Maintenance). Contact information for sales and technical support can be found at:

Avanti Technology, Inc. Web site: www.avanti-tech.com

The latest product release information and updates can also be found on this Web site, as well as sample scripts/tasks and tips, plus technical support FAQs (Frequently Asked Questions).

Index