```
--
-- file: NMConsole.mib
--
-- COMTEK Services, Inc.
-- MIB for VMS NM*Console Subagent
-- Release      3.5
-- Date         January 2002
-- Author       NF
--
-- Copyright 1996-2002 COMTEK Services, Inc.  All Rights Reserved.
--
-- This COMTEK Services VMS NM*Console SNMP Management Information Base
-- Specification (MIB) embodies COMTEK Services' confidential and
-- proprietary intellectual property.  COMTEK Services retains all
-- title and ownership in the Specification, including any
-- revisions.
--
-- This Specification is supplied "AS IS," and COMTEK Services makes
-- no warranty, either express or implied, as to the use,
-- operation, condition, or performance of the Specification.
--

COMTEK-VMS-NM-CONSOLE-MIB DEFINITIONS ::= BEGIN

IMPORTS
     DisplayString                    FROM RFC1213-MIB
     OBJECT-TYPE                      FROM RFC-1212
     TimeTicks, Counter               FROM RFC1155-SMI
     comtekVmsNMConsoleMib            FROM COMTEK-DEFINITIONS-MIB;

cConsole OBJECT IDENTIFIER              ::= { comtekVmsNMConsoleMib 1}
cTrap OBJECT IDENTIFIER                 ::= { comtekVmsNMConsoleMib 2 }
cSwErrInfo OBJECT IDENTIFIER            ::= { comtekVmsNMConsoleMib 3 }
cCfg OBJECT IDENTIFIER                  ::= { comtekVmsNMConsoleMib 4 }


--
cConsoleCmd OBJECT-TYPE
        SYNTAX  DisplayString
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "A command to be executed by the NM*Console Subagent.
                The format of a command is:

                        tag <command>qualifiers

                Where tag is label to be associated with the command,
                <command> is any command that may be entered at the
                DCL prompt (including any parameters or qualifiers) and
                qualifiers is any qualifiers which are to be placed on
                the detached process that is started to execute the
                command.  The following are valid commands:

                        abcd <dir sys$sysdevice:[comtek$snmp]*.com/col=1>

                        test01 <stop/id=1FE>/priv=world
```

```
                "
        ::= { cConsole 1 }

cConsoleTag OBJECT-TYPE
        SYNTAX  DisplayString
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Tag that accompanied a command.  This tag is used
                in each cConsoleResult trap message to associate the
                command results with the command."
        ::= { cConsole 2 }

cConsoleOutput OBJECT-TYPE
        SYNTAX  DisplayString
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Output from a cConsoleCmd command."
        ::= { cConsole 3 }

cConsoleCurrent OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The number of remote console commands currently active."
        ::= { cConsole 4 }

-- The NM*Console Subagent process maintains an internal trap table
-- containing the latest traps that have been sent by the subagent.
-- The number of traps that are retained in the internal trap table
-- is controlled by the cCfgTrapTblSize variable.  Each trap that is
-- sent by NM*Console includes a trap sequence number and a timestamp
-- as well as any trap specific data.  The trap sequence number may
-- be used to request that a trap be resent to the manager.
--
-- The number of traps per second that may be sent is governed by the
-- cCfgMaxTrapSec variable.  cTrapLastSeqNumSent identifies the
-- sequence number of the last trap that was actually sent, if no
-- backlog of traps exists, this number will be one less than
-- cTrapNextSeqNum.  If a backlog exists due to throttling, then
-- the size of the backlog may be determined by comparing these
-- numbers.  If the size of the backlog exceeds the size of the
-- trap table (as set by cCfgTrapTblSize), the backlogged traps
-- will be lost as they are overwritten by new traps, the number
-- of traps that have been lost is contained in cTrapLostCount.

cTrapNextSeqNum OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The next sequence number in the trap table.  When this
                variable is sent as part of a trap, it identifies the
                sequence number assigned to that trap.  This is a 32-bit
                value and will wrap after 4294967295."
```

```
        ::= { cTrap 1 }

cTrapTime OBJECT-TYPE
        SYNTAX  TimeTicks
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "cCfgUpTime when the trap was generated."
        ::= { cTrap 2 }

cTrapResendSeqNum OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "Sequence number of a trap message to be resent to
                the manager.  Setting this variable to a trap sequence
                number causes the specified trap message to be
                resent.  If the set-request contains a sequence
                number that is not available (the number of items
                retained in the trap table is dependent on the value
                of cCfgTrapTblSize), an error indicator will
                be returned."
        ::= { cTrap 3 }

cTrapLastSeqNumSent OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Sequence number of the last trap sent.  This value may be
                compared to cTrapNextSeqNum to determine if a backlog of traps
                exists.  If this value is one less than cTrapNextSeqNum, then
                no backlog exists.  A backlog of traps may exist if more traps
                are generated per second than may be sent (as specified by the
                variable cCfgMaxTrapSec)."
        ::= { cTrap 4 }

cTrapLostCount OBJECT-TYPE
        SYNTAX  Counter
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "This variable indicates the number of traps that have been
                lost.  A trap is considered to be lost if it is overwritten
                in the internal trap table before it is sent to the NMS.
                This may occur if the trap table is not large enough (see
                cCfgTrapTblSize) to hold all of unsent traps.  How quickly
                traps are sent to the NMS is governed by the cCfgMaxTrapSec
                variable.  To prevent trap loss, set these two variables
                so that the trap backlog does not overflow the trap table
                and so that traps do not flood the network."
        ::= { cTrap 5 }

-- When a software error is encountered, a cSwErr trap is generated containing
-- the following software error description items.  Invalid sets of the
-- cCfgFile variable will cause a cSwErr trap to be generated to explain the
```

```
-- error.

cSwErrStatus OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The latest software status code."
        ::= { cSwErrInfo 1 }

cSwErrFile OBJECT-TYPE
        SYNTAX  DisplayString (SIZE (0..255))
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The file name of the software element which generated the
                latest software status code."
        ::= { cSwErrInfo 2 }

cSwErrLineNum OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The line number in the file which generated the latest
                software status code."
        ::= { cSwErrInfo 3 }

cSwErrLastTime OBJECT-TYPE
        SYNTAX  TimeTicks
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "cCfgUpTime when the most recent software status code
                was generated."
        ::= { cSwErrInfo 4 }

cSwErrMessage OBJECT-TYPE
        SYNTAX  DisplayString
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Description of the latest software error."
        ::= { cSwErrInfo 5 }

-- Configuration parameters which are read from disk and may be
-- viewed/changed by the network manager.  All changes to these
-- parameters are written to the disk file.
cCfgFile OBJECT-TYPE
        SYNTAX  DisplayString (SIZE (0..255))
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "Name of the file currently being used for NM*Console
                initialization parameters.  This file is identified by the
                logical name COMTEK$CONSOLE_CONFIG.  Changing this variable
                redefines the COMTEK$CONSOLE_CONFIG logical name.  NM*Console
```

```
                must be reinitialized (by setting the variable
                cCfgReinitSubagent to true) before the new configuration
                file is used.  If a set-request is performed on this variable
                and the specified file name does not exist, is not accessible,
                or contains invalid data, the file name is not changed, an
                error is returned, and a cSwErr trap is generated describing
                the error.

                   Set-requests:  Take effect only after reinitialization
                                  (cCfgReinitSubagent)

                NOTE: To make changes to this value permanent across system
                reboots, the definition of this logical name must also be
                changed in the system startup procedures."
        ::= { cCfg 1 }

cCfgLogFile OBJECT-TYPE
        SYNTAX  DisplayString (SIZE (0..255))
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Name of the log file currently being used by NM*Console.
                If this process is run interactively, this variable will
                have the value SYS$OUTPUT: (standard output)."
        ::= { cCfg 2 }

cCfgReinitSubagent OBJECT-TYPE
        SYNTAX  INTEGER { true(1), false(2) }
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "Flag which may be set to force the NM*Console Subagent to
                reinitialize.  Reinitialization involves resetting the
                process configuration using the parameters contained in
                the cCfgFile and sending the cWarmStart trap."
        DEFVAL  { false }
        ::= { cCfg 3 }

cCfgTraps OBJECT-TYPE
        SYNTAX  INTEGER { enabled(1), disabled(2) }
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "Flag which determines if any trap messages are sent to the
                NM*Master Agent.  Setting this variable to disabled(2)
                disables all traps from being sent.

                   Configuration Variable:  Traps
                   Set-requests:            Take effect immediately

                Changes to this value are written to the cCfgFile."
        DEFVAL  { enabled }
        ::= { cCfg 4 }

cCfgTrapTblSize OBJECT-TYPE
        SYNTAX  INTEGER (10..5000)
        ACCESS  read-write
```

```
        STATUS  mandatory
        DESCRIPTION
                "The maximum number of traps to be retained.  Traps are retained
                for two reasons:  to allow traps to be throttled to prevent
                flooding the network and to allow traps to be resent (see
                cTrapResendSeqNum).  If this value is not sufficiently large,
                traps that are backlogged due to throttling may be lost.  This
                value should be tuned in conjunction with cCfgMaxTrapSec.
                This variable controls how many traps are retained in
                NM*Console's internal trap table.

                   Configuration Variable:  TrapTableSize
                   Set-Requests:            Take effect immediately

                Changes to this value are written to the cCfgFile."
        DEFVAL  { 100 }
        ::= { cCfg 5 }

cCfgHostName OBJECT-TYPE
        SYNTAX  DisplayString (SIZE(0..255))
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The name of the host that the subagent is communicating with.
                This is the host name where the NM*Master Agent resides. This
                variable is not currently used.

                   Configuration Variable:  Host"
        DEFVAL  { "localhost" }
        ::= { cCfg 6 }

cCfgTimeout OBJECT-TYPE
        SYNTAX  INTEGER (0..3600)
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "This variable identifies the time-out value that is to be
                used by the subagent during subtree registration.  If this
                value is set to zero, the default value (saDefaultTimeout)
                contained in the NM*Master Agent will be used.  If the value
                specified exceeds the current maximum that the NM*Master Agent
                will permit (saMaxTimeout), the maximum value will be used.
                This value is specified in seconds.

                   Configuration Variable:  Timeout"
        DEFVAL  { 0 }
        ::= { cCfg 7 }

cCfgMaxConsoleCmds OBJECT-TYPE
        SYNTAX  INTEGER (1..100)
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "This variable indicates the maximum number of remote console
                commands that can be active at one time.

                   Configuration Variable:  MaxCommands
```

```
                   Set-requests:           Take effect at next process
                                           start-up

                Changes to this value are written to the cCfgFile."
        DEFVAL  { 5 }
        ::= { cCfg 8 }

cCfgControlTermProc OBJECT-TYPE
        SYNTAX  INTEGER { enabled(1), disabled(2) }
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Flag which may be used to disable process termination by
                the cCfgTermProc variable.  Setting this variable to enabled(1)
                allows the cCfgTermProc variable to be set to true(1) to
                terminate the NM*Console process.  Setting this variable to
                disabled(2) prevents the cCfgTermProc variable from being set.

                   Configuration Variable:  ControlTermProc"
        DEFVAL  { disabled }
        ::= { cCfg 9 }

cCfgTermProc OBJECT-TYPE
        SYNTAX  INTEGER { true(1), false(2) }
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "Flag which may be used to shutdown the NM*Console process.
                Setting this flag to true causes the NM*Console process to
                respond to the set-request, send the cTermProc trap, close
                the connection to the NM*Master Agent, and shutdown.  If the
                cCfgControlTermProc flag state is disabled(2), the cCfgTermProc
                variable can not be set.

                   Set-requests:           Take effect immediately if
                                           cCfgControlTermProc is enabled(2)"
        DEFVAL  { false }
        ::= { cCfg 10 }

cCfgVersion OBJECT-TYPE
        SYNTAX  DisplayString
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Current version of the NM*Console Subagent software."
        ::= { cCfg 11 }

cCfgUpTime OBJECT-TYPE
        SYNTAX  TimeTicks
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "Subagent up time."
        ::= { cCfg 12 }

cCfgMaxTrapSec OBJECT-TYPE
        SYNTAX  INTEGER (0..100)
```

```
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "This value specifies the maximum number of traps per second
                that may be sent.  A value of zero disables trap throttling,
                allowing all traps to be sent as they are generated.  It
                is recommended that this value and cCfgTrapTblSize be
                tuned to each operational environment.

                   Configuration Variable:  MaxTrapSec
                   Set-requests:            Take effect immediately.

                Changes to this value are written to the cCfgFile."
        DEFVAL  { 0 }
        ::= { cCfg 13 }

END
```