Network Access Server for the Black Box QL

NAServe is a file server for the Black Box QL. It performs the same function as FSERVE in the QL, except that instead of allowing all Network stations full access to the files and devices on the server. NAServe allows you to specify exactly which files and devices may be shared with which Network station.

NAServe has been back ported from the IP Network Device Driver for QL emulators. And also includes some of the Network improvement commands of the IP Network Device Driver.

NAServe contains portions of the SMSQ/E source code. See the end of this document for the copyright licence.

Installing the program

Toolkit 2, or SMSQ/E should be loaded before starting the program.

To load the program into memory and call it.

example: **LRESPR flp1_NAServe_cde**

An installation message, and a version number will be displayed in #0

Note: There is a problem in Toolkit 2 and SMSQ/E up to version 3.32, with the **N8_** device. If the Server is Network station 8, and on a remote station you try to do a directory of a device on the server. e.g. **DIR n8_flp1_**

You will not receive a list of the files on flp1_, just the medium name and the sector counts. The problem also affects all the wildcard commands like **WCOPY** and **WSTAT**.

The problem does not effect normal loading and saving, So **LOAD n8_flp1_program** will operate correctly.

You can get around the problem with the **NFS_USE** command. for example **NFS_USE mdv,n8_flp1_**

DIR mdv1_ will now give the directory of n8_flp1_

The following is an edited and updated extract from the Toolkit 2 manual (c) Tony Tebby

Network Improvements

NET_NUM is a function that will return the network station number.

      PRINT NET_NUM   display the network station number

File Servers

There are now two file servers available. The original **FSERVE** and the new **NASERVE**.

FSERVE allows free remote access to all of the servers devices and files, whereas **NASERVE** provides access controls to the sharing of it's devices and files.

Usage of the two file servers is similar. The use of **FSERVE** will be described first, then the added features of **NASERVE** will be described.

FSERVE

The file server provided in the Network device driver, is a program which allows I/O resources attached to one QL to be accessed from another QL. This means that, for example, disk drives attached to just one QL can be accessed from several different QL's. The file server only needs to be running on the QL with the shared I/O resource. The I/O resources may be pure serial devices (such as modems or printers) or windows on the QL display as well as file system devices (such as disk drives).

      FSERVE   invokes the 'file server'

A client QL may only access up to 8 network file servers at any one time.

It is possible that files opened across the network may be left open. This can occur if a remote QL is removed from the network, is turned off or is reset. To correct this condition, wait until all other remote QLs have finished their operations on this QL, then remove the file server and restart with the commands

      RJOB SERVER
      FSERVE

Accessing the File Server

The network file servers are accessed from remote QL's using a compound device name:

      Nstation number_IO device   the name of a remote I/O device (e.g.
                                           N2_FLP1_ is floppy 1 on network
                                           station 2)

For example

      LOAD n2_flp1_fred   loads file 'fred' from floppy 1 on network
                                           station 2

      OPEN_IN #3,n1_flp2_myfile   opens 'myfile' on floppy 2 on network station 1

      OPEN #3,n1_con_120x20a0x0   opens a 20 column 2 row window on net
                                             station 2

The use of directory default names makes this rather simpler. For example

PROG_USE n1_win1_progs   by default all programs will be loaded from directory 'progs' on Winchester disk 1 on network station 1

SPL_USE n1_ser   set the default spooler destination to SER1 on network station 1

NFS_USE name, network names   sets the network file server name

The 'network names' should be complete directory names, and up to eight network names may be given in the command. Each one of these network names is associated with one of the eight possible directory devices ('name'1 to 'name'8).

Note, the network name can be up to 16 characters in Toolkit 2, and 32 in SMSQ/E

For example

NFS_USE mdv,n2_flp1_,n2_flp2_   sets the network file server name so that any reference to 'mdv1' on this remote QL, will be taken to be a reference flp1 on net station 2, likewise 'mdv2' will be taken to be flp2 on net station 2

OPEN_NEW #3,mdv2_fred   now this will open file 'fred' on floppy 2 on network station 2

The network names will normally just be a network number followed by a device name as above and will end with an underscore to indicate that the name is a directory. Indeed if the network file server name is to be used with the wild card file maintenance commands, this is the only acceptable form. QUILL, however, tends to open a file with the name DEF_TMP on mdv2_. Clearly, there will be problems if more than one copy of QUILL is run across the network at any one time. This can be avoided if the network name for mdv2_ is set to be a directory:

NFS_USE mdv,n1_flp1_,n1_flp2_fred_
    DEF_TMP opened on mdv2_ will now appear in directory 'fred' on flp2_ on network station 1

FLP_USE FLP is invoked after reset so if FLP is to be used as the device name in the **NFS_USE** command remember to include **FLP_USE XXX**.
This will stop the emulator from trying to access its own disk port instead of the network.

FLP_USE xyz   set device name for floppies to xyz

NFS_USE flp,n1_flp1_,n1_flp2_   any reference to 'flp1' on this QL will access flp1 on net station 1, etc.

NFS_USE$ is a function that will return the network file server name, and the network names set using the **NFS_USE** command.

PRINT NFS_USE$(0)   returns the network file server name

PRINT NFS_USE$(2)   returns the second network name

NFS_SET is a command that allows you to individually set the network names for **NFS_USE** without having to reset all of them as with the **NFS_USE** command.

NFS_SET 2,n1_win1_   just sets the second network name without affecting the others

NFS_TYPE is a function that will return the type of file server that is running. If no file server is running, it will return zero. For **FSERVE** it will return 1. And for **NASERVE** it will return minus 1

        PRINT NFS_TYPE   return the file server type

Messaging

The Network device driver network facilities may also be used for messaging. A window may be opened, a message sent, and a reply read using a simple SuperBASIC program. If particularly pretty messages are required, then the graphics facilities of SuperBASIC may also be used. The only standard I/O facilities not available across the network are SD.EXTOP (extended operations) and SD.FOUNT (setting the founts).

For example

        ch = FOPEN (n2_con_150x10a0x0): CLS #ch
        INPUT #ch,'Do you want coffee? ';rep$
        IF 'y' INSTR rep$ = 1 : PRINT 'Fred wants coffee'
        CLS #ch: CLOSE #ch

NASERVE

While **FSERVE** allows all network stations full access to the devices and files on the file server. **NASERVE** (Network Access SERVEr) will by default, block all other network stations access to it's devices and files.

Devices and files on the Network Access Server must be specifically shared, either globally across the network, or on an individual station by station basis.

Devices and files on the Network Access Server are shared using the **NAS_SHARE** command, and unshared by the **NAS_UNSHARE** command. Additionally a list of the currently set shared devices and files can be displayed with the **NAS_SHARED** command.

The **NAS_SHARE** command can have an additional parameter added to set a directory device, a file, or a sub-directory as read only to the remote network stations.

While it is possible to flag non directory devices as read only, it does not always make sense and should be avoided.

Care should be taken when entering a list of network shares, as **NASERVE** checks network share requests in the order in which the **NAS_SHARE** commands were entered.

For example:
Suppose you want to share the file servers **win1_shared_** directory with everyone, except for network station 4, which will have read only access.
If you enter

        NAS_SHARE 0,win1_shared_
        NAS_SHARE 4,win1_shared_,1

This would allow network station 4 to write to **win1_shared_** Because when **NASERVE** checks the **OPEN** command from network station 4, it will first see the share **0,win1_shared_** . Network station 4 comes under everyone, so it will allow network station 4 to write to **win1_shared_**

However if you enter:

        NAS_SHARE 4,win1_shared_,1
        NAS_SHARE 0,win1_shared_

Then **NASERVE** will first check the share **4,win1_shared_** which will match, and it will give network station 4 the read only access required.

Note also that the **DEV** device could also cause possible problems.
If you enter

DEV_USE 6,ram1_
NAS_SHARE 2,ram1_,1
NAS_SHARE 0,ram1_
NAS_SHARE 0,dev6_

So that **ram1_** is shared with full access to all stations, except station 2, which is read only.
However, station 2 will have full access to **ram1_** via **dev6_**

You would need to add an extra **NAS_SHARE** to stop station 2 writing to **dev6_**

DEV_USE 6,ram1_
NAS_SHARE 2,ram1_,1
NAS_SHARE 0,ram1_
NAS_SHARE 2,dev6_,1
NAS_SHARE 0,dev6_

The **NAS_SHARED** command lists all of the currently shared resources.

It will display a list in the format of Router station number, Network station number, Resource name, Access type. Where the access type is 0 for read/write access, and 1 for read only access.

The Router station number is an unused feature in NAServe, and is left in to provide compatibility with the IP Network Device Driver. You can set the Router station number by multiplying it by 256, and adding it to the station number in the **NAS_SHARE** command.

NAS_SHARE 4*256+2,flp1_

Would set the Router station number to 4, and the station number to 2. The NAServe file server will ignore the Router station number

When a remote network station tries to open a device or file. The order that **NASERVE** checks the requested **OPEN** command against each of the share entries, is as follows:

1. The requesters station number, unless the share entries station number is 0

2. Compares the names

3. Checks the read only status

The only **OPEN** commands that **NASERVE** will allow through for a read only access device or file, is **OPEN_IN** and **OPEN_DIR**

Operation of the **FSERVE** fileserver is unaffected, however **FSERVE** and **NASERVE** may not be used at the same time. If you attempt to start one server type, while the other type is already running, you will receive an 'is in use' error.

NET_NUM

NET_NUM is a function which returns the network station number as an integer.

It will return the number entered in the **NET** command, or 1 (the default Network station number) if the **NET** command has not been used.

syntax: **NET_NUM**

example: **myStationNumber = NET_NUM**

FSERVE

The **FSERVE** command is used to start the file server.

syntax: **FSERVE**

example: **FSERVE**

NASERVE

The **NASERVE** command is used to start the share file server.

If you attempt to start **NASERVE** while **FSERVE** is already running, you will receive an 'is in use' error.

If **FSERVE** is running it may be stopped with a **RJOB SERVER** command

syntax: **NASERVE**

example: **NASERVE**

NAS_SHARE
FNAS_SHARE

The **NAS_SHARE** command is used to add a device or file to the list of shared resources.

A network station number defines the station that can use this device or file, Specifying a station number of 0, shares the resource for all users.

An optional access type defines whether the device or file has read/write or read only access. 0 (the default) defines that the device or file has read/write access, and a positive number defines the device or file has read only access.

Note that a read only protected device or file can only be accessed by an **OPEN_IN** or an **OPEN_DIR**

If you just want to change a shared device or file access type, it is not necessary to unshare, and then re-share the device or file again. Just share it again with the same network station number and device/file name.

FNAS_SHARE is a function version of **NAS_SHARE** that returns zero, or an error code, without stopping the running program.

syntax: *station_number* := *numeric_expression*   {0 to 254}
            resource := *device/file name*
            access_type := *numeric_expression*

            NAS_SHARE *station_number*,*resource*,[*access_type*]
            FNAS_SHARE (*station_number*,*resource*,[*access_type*] **)**

example: i.   **NAS_SHARE 0,ram1_**   {share ram1_ with everybody}
           ii. **NAS_SHARE 1,win1_shared_**   {share win1_shared_ with net 1}
           iii. **NAS_SHARE 1,win1_shared_,1** {make win1_shared_ read only to net 1}
           iv. **NAS_SHARE 3,con_**   {allow net 3 to open consoles}
           v.   **NAS_SHARE 2,ser1**   {allow net 2 use my ser1 port}
           vi. **result=FNAS_SHARE (2,flp1_)** {allow net 2 to use my flp1_}

Note: When **NAS_SHARE** is used. Entries are added in the order that they are supplied. Unless you are changing the access type of an existing entry.

NAS_UNSHARE
FNAS_UNSHARE

The **NAS_UNSHARE** command will remove a shared device or file from the share list.

NAS_UNSHARE without parameters will remove all the shared permissions.

FNAS_UNSHARE is a function version of **NAS_UNSHARE** that returns zero, or an error code, without stopping the running program.

syntax: *station_number* := *numeric_expression*   {0 to 254}
            resource := *device/file name*

            NAS_UNSHARE *station_number,resource*
            FNAS_UNSHARE (*station_number,resource* **)**

example: i. **NAS_UNSHARE 1,win1_shared_**   {stop sharing win1_shared_ with net 1}
           ii. **NAS_UNSHARE 0,ram1_**   {stop sharing ram1_ with everybody}
           iii. **NAS_UNSHARE**   {remove all shares}
           iv. **result=FNAS_UNSHARE(2,flp1_)**   {stop sharing flp1_ with net 2}

Comment: Un-sharing a device or file which already has an open channel, will have no effect on that connected channel as sharing rights are only tested during **OPEN**s.

NAS_SHARED

The **NAS_SHARED** command is used to display a list of the currently shared resources.

It will display a list in the format of Router station number, Network station number, Resource name, Access type. Where the access type is 0 for read/write access, and 1 for read only access.

An optional Basic channel number may be specified. With channel #1 being the default.

syntax: *channel_number* := *numeric_expression*

            NAS_SHARED [#*channel_number*]

example: i. **NAS_SHARED**   {send shared list to Basic channel 1}
           ii. **NAS_SHARED#2**   {send shared list to Basic channel 2}

NFS_USE

The **NFS_USE** command sets the network file servers name, so that any reference to the supplied 3 letter device name on this remote QL, will be taken to be a reference to one of the supplied server devices.

The network names should be complete directory names, and up to eight network names may be given in the command. Each one of these network names is associated with one of the eight possible directory devices (name1_ to name8_).

syntax: *name* := *3_letter_device _name*
             network_name := *server_device*

        NFS_USE *name*, *network_names*   {up to 8 network names}

example: i.   **NFS_USE mdv,n2_flp1_,n2_flp2_**   {mdv1_ will reference n2_flp1_ and
                                                              mdv2_ will reference n2_flp2_}
           ii. **NFS_USE mdv,n1_win1_,n2_flp1_,n1_ram1_**
                COPY mdv2_fred TO mdv3_fred   {copy the file fred from flp1_ of server
                                                            2, to ram1_ of server 1}

comment: The network names will normally just be a network number followed by a device
           name, and will end with an underscore to indicate that the name is a directory.
           Indeed if the network file server name is to be used with the wild card file
           maintenance commands, this is the only acceptable form. QUILL, however, tends to
           open a file with the name DEF_TMP on mdv2_. Clearly, there will be problems if
           more than one copy of QUILL is run across the network at any one time. This can be
           avoided if the network name for mdv2_ is set to be a directory:

           NFS_USE mdv,n1_flp1_,n1_flp2_fred_

           DEF_TMP opened on mdv2_ will now appear in directory 'fred' on flp2_ on network
           station 1

NFS_USE$

NFS_USE$ is a function which returns the name, or the network names entered in a **NFS_USE** command.

The entry number parameter determines which network name is returned, with entry number 0 being the **NFS_USE** name.

If the name, or the required network name is missing. **NFS_USE$** will return an empty string.

syntax: *entry_no* := *numeric_expression*

        NFS_USE$(*entry_no***)**   {0 - 8}

example: i.   **PRINT NFS_USE$(0)**   {the **NFS_USE** name will be displayed}

        ii.   **a$=NFS_USE(2)**   {a$ is set to the second network name}

comment: If the command **NFS_USE mdv,n1_win1_,n2_flp1_,n1_ram1_** has been used, then example (i) above will return '**mdv**'. And example (ii) will return '**N2_flp1_**'.

NFS_SET

The **NFS_SET** command allows any of the eight network names set by the **NFS_USE** command to be individually changed without having to reset all of the network names at the same time with as with **NFS_USE** command.

syntax: *entry_no* := *numeric_expression*   {1 – 8}
          network_name := *server_device*

        NFS_SET *entry_no*, *network_name*

example: **NFS_SET 2,n2_flp2_**   {change the second entry that was set with **NFS_USE**}

comment: If you wish to change the device name used by **NFS_USE**, just use the **NFS_USE** command with one parameter e.g. **NFS_USE nfa**.

NFS_TYPE

NFS_TYPE is a function that will return the type of file server that is running.

If no file server is running, it will return zero. For **FSERVE** it will return 1. And for **NASERVE** it will return minus 1

syntax: **NFS_TYPE**

example: **PRINT NFS_TYPE**

Copyright and Disclaimer

This driver should not cause any problems, damage, or loss of data. However by using this device driver, you do so at your own risk, and I do not accept responsibility for any damage, or loss of data.

<u>Licence for SMSQ/E</u>

Copyright (c) 1989-2012, by

> Tony Tebby
> Marcel Kilgus
> Bruno Coativy
> Fabrizio Diversi
> Phoebus Dokos
> Thierry Godefroy
> Jérôme Grimbert
> George Gwilt
> John Hall
> Mark Swift
> Per Witte
> Wolfgang Lenerz

collectively called the "COPYRIGHT HOLDERS".

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

 * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
 * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL COPYRIGHT HOLDERS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.