Network Application Installer

N.A.I.

Release 1.5

7319 Willow Avenue

Takoma Park, MD 20912

TABLE OF CONTENTS

TABLE OF CONTENTS, Continued

PREFACE:

What's New in Release 1.5

For those users who have purchased or evaluated earlier versions of N.A.I., here is a list summarizing the major enhancements in release 1.5:

User-specific variables

N.A.I. now fully supports the use of user-specific variables, within the WIN.INI and SYSTEM.INI files in destination paths for files to copy, and in the command line, icon path, and working directories.

SYSTEM.INI modification

N.A.I. now provides a means of automatically modifying information within the SYSTEM.INI file. This function works just like the modifications to WIN.INI.

Icon Additions to Existing Icon Groups

Release 1.5 adds a third method of installing icons: adding them to existing icon groups.

NAI Installation Log

N.A.I. now maintains its own log of installed applications on each local PC, in the file NAI.INI. This file contains the time and date of each application that N.A.I. installs; it is updated each time an application is installed or removed.

Improved User Interface

The user interface to the end user program, NAI.EXE, has been upgraded substantially to provide a more pleasing appearance. The user interface to the maintenance program, NAIMAINT.EXE, has also been improved, though not as dramatically.

Miscellaneous Minor Enhancements and Fixes

Release 1.5 contains numerous minor improvements, including automatic creation and removal of directories and nested subdirectories, increasing the allowable size of WIN.INI additions from 100 to 350 lines per .DAT file, increasing the allowable size of additions to the OLE and Embedding sections of WIN.INI from 20 to 50 lines each, enabling the WIN.INI Additions Editor to paste existing data from the clipboard into the WIN.INI Additions section of a .DAT file, the use of a listbox of existing .DAT files for the *INSERT APPLICATION INTO LIST* function, and a number of other small corrections and enhancements.

OVERVIEW:

What the Network Application Installer (N.A.I.) Is

One of the great advantages of a PC LAN file server is that administrators can provide their entire user community with access to a software program simply by installing a single copy of the program on the server. Unfortunately, many Microsoft Windows applications are not amenable to this approach, because Windows programs often require specific changes to the environment of each PC where the application is to run. For example, a Windows application install routine is likely to make changes to the user's WIN.INI file, to create special, application-specific .INI files, to create special .DLL files, etc. The application may not run, or may not run properly, without these changes. Until now, the only means of meeting the need for these local configuration changes was to make them manually to each PC on the LAN or else to run the application install routine on each PC on the LAN.

The Network Application Installer, N.A.I., eliminates these problems, allowing a LAN administrator to install a Windows application only once, and providing end users on the LAN with the means of installing in their local workspace (or removing from their workspace) any Windows application with a click of the mouse. The convenient operation of N.A.I. makes quick work of adding new applications or upgrading installed applications to the latest release.

N.A.I. was initially designed to enable users to install applications which would run from the file server, but it can work equally well as a method of distributing full applications to each PC's local hard disk.

In addition, whether you run your Windows applications locally or from the file server, N.A.I.'s removal option provides end users with the most effective tool available to help keep their local Windows configurations up to date and free of extraneous, leftover files and information. This option alone may be reason enough to install N.A.I. on your file server.

An important feature of N.A.I. is that it provides the ability to install *any* Windows program, and *an unlimited number of Windows programs*, without purchasing any additional modules or waiting for their development. You simply install an application once, note the modifications it makes, and then create the necessary N.A.I. module yourself in minutes, using the N.A.I. Maintenance Program from within Windows! Your users are instantly provided with the means of installing the new application at the click of a mouse.

Although some new applications now include a node install program which makes the necessary configuration changes on the local workstation, N.A.I. provides the end user with superior ease of use, plus the ability to remove the application from the local configuration at any time.

When N.A.I. installs an application, it performs three critical actions automatically:

1. Modifies WIN.INI and/or SYSTEM.INI as required.

2. Copies any needed files to the local disk.

3. Installs the appropriate icons in the local workspace.

Some applications may require as little in the way of local configuration changes as the single act of placing an icon in the local workspace, but N.A.I. is helpful even with these simple installations because it relieves the end user of having to know or enter anything concerning directories or executable file names; the icon can be installed in the workspace by a simple click of the mouse with no chance of typographical errors or other miscues.

How N.A.I. Works

N.A.I. is really two programs: NAI.EXE, which the end user runs to install or remove programs, and NAIMAINT.EXE, which the LAN administrator runs to configure NAI.EXE to make applications available for automatic installation and removal.

NAI.EXE

When a user runs N.A.I., the program NAI.EXE reads (from the file WINAPPS.LST) the list of applications available to be installed or removed. WINAPPS.LST also tells N.A.I. whether or not the *REMOVE APPLICATION* function is enabled, allowing users to remove applications as well as to install them. When the user selects an application to install or remove, N.A.I. reads the data file (filename.DAT) for that application. The .DAT file tells N.A.I. what changes to make to the user's local configuration for the installation or removal of that application. N.A.I. then automatically makes the changes, including placing the icon on the user's workspace, and lets the user know that the installation was successful. A typical application can be completely installed in this manner in well under 10 seconds!

NAIMAINT.EXE

Four types of data files are associated with the N.A.I. programs:

WINAPPS.LST (The list of applications

available for installation or removal.)

Application .DAT files (Instructions to NAI.EXE for installation and removal of a particular application)

NAIMAINT.CFG (the maintenance program configuration file)

NAI.INI (the N.A.I. log file)

All are ASCII text files, but each has its own specific format. To avoid requiring the LAN administrator to understand and remember the details of those formats (although they are fully described later in this document), and to avoid inadvertent errors in file formats, all these files are created and maintained by means of an additional program: NAIMAINT.EXE (except for NAI.INI, which is created maintained automatically by the end user program, NAI.EXE).

NAIMAINT.EXE allows the LAN administrator to create new application .DAT files and modify or delete existing application .DAT files, at the same time updating WINAPPS.LST, the list of available applications that users see when they run NAI.EXE. NAIMAINT.EXE also permits the LAN administrator to enable or disable the *REMOVE APPLICATION* function in NAI.EXE, and to invoke the use of user-specific variables, if desired.

INSTALLATION:

User Files

The user files (NAI.EXE, WINAPPS.LST, and all the application .DAT files) should reside in the same directory on the file server. Users need not have write access to this directory.

Each user must have the file VBRUN100.DLL in the Windows directory and the N.A.I. icon in the workspace.

LAN Administrator Files

NAIMAINT.EXE and NAIMAINT.CFG can reside in the Network Application Installer (N.A.I.)
same directory as the user files, or, for better security, they can reside in another directory. (The *CURRENT WORKING DIRECTORY* field on the main screen of NAIMAINT.EXE allows the LAN administrator to specify the location of the user files). The LAN administrator must have read, write, and delete privileges for the directory where the user files reside and for the directory where the maintenance files reside.

The LAN administrator must have VBRUN100.DLL in his local Windows directory and the NAIMAINT icon in his workspace.

PROGRAM OPERATION:

Choosing an Application to Install

When the user clicks on the Network Application Installer icon, the NAI.EXE program reads the file WINAPPS.LST, which lists all the applications available for installation. To install an application, the user simply clicks once on the desired application in the list and then clicks on the *INSTALL APPLICATION* Button.

If N.A.I. discovers that the user has already installed this application, it will offer a choice of canceling the installation or performing a re-installation.

The final step in the installation is the placing of the application icon(s) in the user's Windows workspace. If the application .DAT file specifies the creation of a separate icon group, N.A.I. will create that group and place the icon(s) in it; if the application specifies that the icons be added to an existing group, N.A.I. will do that, creating the group if it does not already exist; otherwise, it will place the icons in whatever is the active group on the end user's workspace at the time of the installation.

Choosing an Application to Remove

To remove an application, the user clicks on the application in the application list and then clicks the *REMOVE APPLICATION* Button. At this point, N.A.I.

reads the application .DAT file and then removes the appropriate WIN.INI lines, files, etc., from the user's local configuration.

If the application .DAT file specifies that the icons for the application belong in a separate icon group, N.A.I. will remove the group and any icons it contains from the end user's local workspace. On the other hand, if the application .DAT file specifies that the icons belong in an existing group or in whatever the active group is at the time of installation, N.A.I. will not remove any icons from the user's workspace; instead, at the end of the removal process, N.A.I. will advise the user that the icon may not have been removed and will provide simple instructions on how to do so manually.

PROGRAM MAINTENANCE:

Maintenance Program Startup

When the LAN administrator clicks on the NAIMAINT.EXE icon, the program looks for the file NAIMAINT.CFG in the NAIMAINT.EXE program directory. If this file is found, NAIMAINT.EXE reads it to discover where the correct version of the file WINAPPS.LST is located. If NAIMAINT.CFG is not present, then NAIMAINT.EXE looks for WINAPPS.LST in the NAIMAINT.EXE program directory. If WINAPPS.LST is not found, NAIMAINT.EXE informs the LAN administrator, but proceeds anyway, providing the opportunity to create a new WINAPPS.LST or to change the working directory to one which does contains a WINAPPS.LST file.

NAIMAINT.EXE reads the file WINAPPS.LST to build its list of installable applications before displaying the initial screen. This screen offers the options of setting the working directory, enabling or disabling the N.A.I. REMOVE option, setting user-specific variables, plus several choices regarding the list of available applications: create a new application entry in the list, modify an existing application entry, insert an application into the list, or remove an existing application from the list.

Setting the Working Directory

To enable the security of keeping NAIMAINT.EXE in a directory separate from the NAI.EXE program, the file NAIMAINT.CFG, which NAIMAINT.EXE reads on startup, is kept in the same directory as NAIMAINT.EXE. This file stores the location of the last working directory, the directory where NAI.EXE, WINAPPS.LST, and the application .DAT files are kept. If this configuration file is missing, NAIMAINT.EXE sets the working directory to the directory where NAIMAINT.EXE itself is located.

The current working directory is displayed in a modifiable field on the NAIMAINT.EXE main screen. When that directory is changed, NAIMAINT.EXE immediately switches to the specified directory and loads the WINAPPS.LST file from that directory, if there is one. If it does not find a WINAPPS.LST file in the specified directory, it informs you and begins by presenting a blank list of applications. Any changes will result in the creation of a new WINAPPS.LST in whatever the working directory is at that time; the actual file will be created either when the NAIMAINT.EXE program is exited, or when the working directory is changed.

Enabling the REMOVE Option

On the NAIMAINT.EXE main screen is a pair of radio buttons which enable or disable the *REMOVE APPLICATION* option in NAI.EXE. If this option is toggled in NAIMAINT.EXE, then a new WINAPPS.LST file will be created on exit from NAIMAINT.EXE, whether or not any changes were made to the application list itself. When a new WINAPPS.LST file is created (on exit from NAIMAINT.EXE or when the working directory is changed), the current setting of the *REMOVE* radio buttons determines whether or not the *REMOVE APPLICATION* option in N.A.I. is enabled or disabled.

When the *REMOVE APPLICATION* option is disabled, N.A.I. users will not see the *REMOVE* Button on their screen and will not be able to use NAI.EXE to remove from their local configurations any of the listed applications.

Setting the User-Specific Variables

N.A.I. supports the use of up to three separate user-specific variables during the installation and removal of applications. User-specific variables allow installations to vary according to information that may be unique to each user. For example, if you want to install certain files for an application in a directory that is named differently for each user, N.A.I.'s user-specific variable feature enables you to do so.

If you choose to implement user-specific variables, N.A.I. will prompt each user at runtime for the unique information you need. After receiving this information from the user, N.A.I. will use it during the installation or removal of applications by substituting the unique, user-specific information for special codes ($VAR codes) it encounters in the application .DAT files.

User-specific variables are implemented in two steps: First, you must instruct N.A.I. what information to request from the user. Second, you tell N.A.I., in each application .DAT file, what to do with that information.

To tell N.A.I. what information you need from the user, click on the *SET USER-SPECIFIC VARIABLES* Button on the NAIMAINT.EXE Main Screen. The User-Specific Variables screen will appear, with three blank data entry screens labeled *PROMPT 1, PROMPT 2*, and *PROMPT 3.* You may enter information in any or all of these fields. For example, if you wanted to place certain application files in a directory named according to the user's network username, you could enter for *PROMPT 1*, the following text:

Your Network Username

If information has been entered for *PROMPT 1*, *PROMPT2*, or *PROMPT3*, then when an end user runs N.A.I. and selects the *INSTALL APPLICATION* or *REMOVE APPLICATION* option for the first time, N.A.I. will present a screen asking for the unique, user-specific information you specify, prompting him with whatever text you enter in the *PROMPT 1, PROMPT 2*, and/or *PROMPT 3* fields.

NOTE: Use of the user-specific variables feature is entirely optional: N.A.I. will only prompt for those fields where you have actually entered information, and if you do not enter information for any of the fields, N.A.I. will not request any information from the users and will not attempt to implement any user-specific installation parameters.

To continue the example above, if you entered the text shown above, then N.A.I. would present a screen instructing the user as follows:

N.A.I. needs the following information in order to proceed:

Your Network Username: ________________________________

When the user enters the requested information, N.A.I. will proceed with the installation or removal, substituting the text supplied by the user for the corresponding $VAR codes encountered within the application .DAT file.

Thus, the second step in implementing user-specific variables is to place the $VAR codes into the application .DAT files as required. N.A.I. will interpret $VAR codes wherever it finds them within text to be added to the WIN.INI and SYSTEM.INI files, within directories and filenames specified in the FILES TO COPY fields of the Add/Modify Application screen, and in the directories and filenames in the ICONS TO COPY fields. Each of these areas is explained in detail in its own section, below.

Each of the three available user-specific variables has a different $VAR code. For the variable corresponding to *PROMPT 1*, you would enter *$VAR1$* (note both the leading and trailing *$* characters) in the application .DAT file. For the variables corresponding to *PROMPT 2* and *PROMPT 3*, you would enter *$VAR2$* and *$VAR3$*, respectively.

To continue the example, if you had instructed N.A.I. to prompt the end user for his network username, and he had entered *BWILSON* in response, then N.A.I. would substitute *BWILSON* for the $VAR code *$VAR1$* whenever that code appears during the installation. So, if you had indicated that a file should be copied to the destination directory *F:\$VAR1$*, N.A.I. would copy the files to the *F:\BWILSON\* directory, creating it in the process, if necessary.

Likewise, if you had the line *DOC-PATH=F:\$VAR1$\DOCS* in the WIN.INI Additions section of an application, what N.A.I. would actually add to the user's WIN.INI file would, in the case of our example user, be this:

DOC-PATH=F:\BWILSON\DOCS\.

N.A.I. will only prompt the user for this information once during each execution of NAI.EXE. If the user installs several programs without exiting NAI.EXE, the program will remember the responses he provided for the first installation and use them whenever an application .DAT file contains a $VAR code. If the user exits NAI.EXE, however, the information he provided is not retained and must be re-entered when an application is installed or removed during subsequent executions of NAI.EXE.

Creating a New Application Entry or Modifying a Listed Application Entry

To create a new application entry in the list of available applications, simply click on the *CREATE NEW APPLICATION* Button. At this point, NAIMAINT.EXE brings up the Add Form, which contains 4 blank data entry fields and three data entry buttons.

To modify the entry for a listed application, click on the desired application in the list and then click on the *MODIFY LISTED APPLICATION* Button. NAIMAINT.EXE will bring up the Modify Form, which is identical to the Add Form, except that the data entry fields will display whatever information was found in the application .DAT file for the application you have chosen to modify.

DATA ENTRY FIELDS

N.A.I. List Name

In the first field on the Add/Modify Form, *N.A.I. LIST NAME,* enter the name of the application exactly as you would have it appear in the N.A.I. application list.

.DAT File Name

The next field, *.DAT FILE NAME,* holds the DOS filename of the file which N.A.I. will use to store the changes needed to install this application. The extension *.DAT* will automatically be appended to the name entered here.

Files to Copy

If a Windows application requires that any files be present on the user's local hard disk, these should be specified in the *FILES TO COPY* fields. Enter in the *SOURCE* field the filename, including drive and full path, of the file to be copied to the local disk. DOS wild cards (* and ?) are supported, but please read the discussion of **Wild Card Deletions** in the **Reference Notes** section of this document for important implications of using wild cards.

If drive and/or path is omitted from the *SOURCE* field, N.A.I. will look only in the N.A.I. directory itself.

If a source file is not found during installation, N.A.I. will inform the user of that fact and abort the installation.

When you move the cursor or mouse pointer from the *SOURCE* field, NAIMAINT will check to see if the file you have specified is, in fact, found where you have indicated it should be. If the file is not found, NAIMAINT will inform you of that fact and ask whether you want to continue or not.

Enter in the *DESTINATION* field the full drive and path of the local copy of the file that N.A.I. will create at installation time (**NOTE:** omit the filename--N.A.I. will automatically create the destination file with the same name as the source file).

If drive and/or path is omitted from the *DESTINATION* field, N.A.I. will default to the drive where Windows is located and to the Windows directory.

If a specified destination directory is not present on the user's local hard disk, N.A.I. will create it automatically during the installation.

If you need to place files in a directory or path that is unique to each user, you can do so by making use of N.A.I.'s user-specific variable feature. If N.A.I. encounters the codes *$VAR1$*, *$VAR2$*, or $*VAR3$* anywhere in a destination path, then during the installation or removal process, it will substitute for that code whatever information it has obtained from the user. For a complete explanation of this feature, please see the **User-Specific Variables** section, above.

NOTE: If the *DESTINATION* field is left blank (or if you enter *@WINDOWS* as the destination directory), N.A.I. will copy the file(s) to the user's Windows directory (for example, C:\WINDOWS), whatever it may be. If you enter *@SYSTEM* in the *DESTINATION* field, N.A.I. will copy the file(s) to the user's Windows System directory (for example, C:\WINDOWS\SYSTEM), whatever that may be.

The *SOURCE* and *DESTINATION* fields are arranged as two data entry fields, each above a list. The two lists are linked. Once a source and destination have been entered, they will drop together down into their respective lists. To add another pair, click on either data entry field and type in the desired information, or click the *ADD* Button. Once both fields have been completed, they are added to the lists automatically when a TAB or mouse click passes the focus to another object or field on the screen, when the *ADD* Button is clicked, or when the *OK* Button is clicked.

To delete a file from the list, click on either the source or destination list to select the desired pair, then click on the *REMOVE* Button.

To modify an existing source and destination pair, either select the pair by clicking on either one in the list, and then click on the *CHANGE* Button, or double-click on either entry in its list. Either method will remove the pair from their lists and place them in the data entry fields above, where they can be modified and added once again to the lists. As with adding new information, the modified information is added to the lists when a TAB or mouse click passes the focus to another object or field, when the *ADD* Button is clicked, or when the *OK* Button is clicked.

WIN.INI ADDITIONS BUTTON

When the *WIN.INI ADDITIONS VIEW/EDIT/LOAD* Button is clicked, NAIMAINT.EXE displays the WIN.INI Additions Editor. In the editor, you can type in or load any changes to the WIN.INI file which must be applied to a user's configuration to install the application.

If the *VIEW/EDIT/LOAD* Button is clicked during a *MODIFY* operation, or if WIN.INI additions have been entered earlier during an *ADD* operation, the WIN.INI Editor will display the changes and allow their further modification.

This editor is a full ASCII editor with buttons to perform the following functions:

· Load text from a file
· Append text from a file
· Save text to a file
· Cut selected text to the Windows Clipboard
· Paste text from the Clipboard
· Delete all text in the editor
· Delete all text in the editor, except selected text
· Quit without saving changes
· Exit and save the changes in the application .DAT file.

The text you enter in the WIN.INI Additions Editor can have embedded in it the codes *$VAR1$*, $*VAR2$*, and/or $*VAR3$*, which instruct N.A.I. to substitute in their place whatever user-specific information it has obtained from the end user at runtime. More information on this option is provided in the **User-Specific Variables** section, above.

SYSTEM.INI ADDITIONS BUTTON

When the *SYSTEM.INI ADDITIONS VIEW/EDIT/LOAD* Button is clicked, NAIMAINT.EXE displays the SYSTEM.INI Additions Editor. This editor functions exactly like the WIN.INI Additions Editor, except that any changes you type in or load here will be applied to the user's SYSTEM.INI file when N.A.I. installs the application. The SYSTEM.INI Additions Editor will also substitute user-specific information for the $*VAR* codes at runtime.

ICONS TO INSTALL BUTTON

N.A.I. needs three to five pieces of information in order to install an icon on the user's workspace. Pressing the *ICON VIEW/EDIT* Button brings up the Icons Screen, where this information can be entered.

Icon Group Radio Buttons

At the top of the screen is a group of three radio buttons permitting you to select where you want the icons for this application to be installed. By default, the *ADD TO CURRENT (ACTIVE) ICON GROUP* Button is selected. This choice will instruct N.A.I. to place the icons in whatever Program Manager group is active at runtime. The user can then move that icon to another group or leave it where it has been placed.

Selecting the *ADD TO EXISTING ICON GROUP* Button will cause a data entry field to appear beside the radio buttons. In this field, enter the name of the existing icon group where you want the icons placed. If N.A.I. does not find a group by this name in the user's configuration, it will create it.

NOTE: The icon group name is the text which appears in the icon group title bar on the Windows workspace.

To install all the icons for an application in their own separate icon group, click on the *CREATE SEPARATE ICON GROUP* Button and enter the desired group name in the data field which appears beside the radio buttons. N.A.I. will create this icon group in the user's configuration at install time, unless the group already exists, in which case N.A.I. will simply add the icons to this existing group.

NOTE: When N.A.I. is REMOVEing an application, its treatment of the icons varies, depending on which icon group radio button is selected for that application. If the application .DAT file instructs N.A.I. to create a separate icon group, then N.A.I. will delete that entire icon group when it removes the application. Otherwise, N.A.I. will NOT remove any icons; instead, it will conclude the removal process with a message to the user indicating that the icons may still be present in his workspace and providing simple instructions on how the user can remove those icons manually.

Because some Windows applications install more than one icon, the information required for installation of icons is presented in three fields which, like the *SOURCE* and *DESTINATION* Fields for *FILES TO COPY,* are presented as data entry fields above linked lists:

Description

In the *DESCRIPTION* Field, enter the text that you would like to appear below the icon in the user's workspace.

Command Line

In the *COMMAND LINE* Field, enter the drive and full path and filename of the executable file for the application. You may use the $*VAR* codes within this field to instruct N.A.I. to substitute user-specific information at runtime. For further details, see the **User-Specific Variables** section, above.

Icon File Name

If the icon to be used for the application comes from other than the application's executable file, enter in the *ICON FILE NAME* Field the drive and full path and filename of the file from which to extract the icon. If this field is left blank, NAIMAINT.EXE will automatically fill it in with the path and filename of the application's executable file. You may use the *$VAR* codes within this field, too, to instruct N.A.I. to substitute user-specific information at runtime. For further details, see the **User-Specific Variables** section, above.

Working Directory

Beginning with release 3.1, Windows offers the option of specifying a working directory for each application. This directory is where the application will by default look for and save its documents or data files. When a user runs N.A.I., the program checks to see what version of Windows the user is running. If the version is 3.1 or above, N.A.I. will install the directory entered in this field as the application's working directory. If the version is 3.0, N.A.I. will ignore any information in this field. As with the *Command Line* and *Icon File Name* fields, you may use the *$VAR* codes within this field to instruct N.A.I. to substitute user-specific information. For further details, see the **User-Specific Variables** section, above.

The same techniques for adding, removing, and modifying pairs of source filenames and destination directories apply to the sets of icon descriptions, command lines, icon file names, and working directories (see the **Files to Copy** section, above). The only differences are that the icon file name, if left blank, will automatically default to the path for the executable file name, and the working directory may be left blank if desired.

When these fields are complete, clicking the *OK* Button saves the changes and returns the Add/Modify Form.

When the fields on the Add/Modify Form have been filled in and the WIN.INI Changes and Icon Information have been entered, click the *OK* Button to create the new application .DAT file and update the list of applications.

Inserting an Application into the List

When you click on the *INSERT APPLICATION INTO LIST* Button, N.A.I. presents a blank data field and a list box of application .DAT files in the Current Working Directory. This feature enables you to add an existing .DAT file to the list. For example, if an application .DAT file had been created in another directory, it could be copied to the current working directory and it and its application added to WINAPPS.LST by using the *INSERT APPLICATION INTO LIST* Button. The text entered into the first field will appear in the *AVAILABLE APPLICATIONS* listing in N.A.I. (and also in the NAIMAINT.EXE program), while the .DAT file selected from the list will provide N.A.I. with the information it needs to install or remove that application from the user's configuration.

SUGGESTIONS:

N.A.I. Configuration

You might consider running several copies of N.A.I. on the same file server. (This is a no-extra-cost configuration option, because N.A.I. is licensed by the file server, not by the copies in use or by the number of nodes or users). For example, you might want to make certain applications available to everyone but other applications available only to certain users. To accomplish this, you could create two directories containing N.A.I., each with a different WINAPPS.LST appropriate to the groups of users having access to it.

Using two directories, two copies of NAI.EXE, and two separate WINAPPS.LST files would also allow you to control which applications users can remove and which they cannot remove; you would simply enable the *REMOVE APPLICATION* option in one WINAPPS.LST file but not in the other.

.DAT File Creation

Lately, some new Windows programs have included a file which describes all changes made by the installation process. These files make creating a .DAT file very easy.

In absence of such a file, it may be easiest to create a brand new Windows configuration on a drive which does not have one (or on a PC which does not have one, if you have the luxury of being able to reserve a PC for this type of work), and install the application onto the file server from that configuration. If you make a backup of the WIN.INI and SYSTEM.INI files and a printout of the WINDOWS, SYSTEM, and ROOT directories before installing the application, you can quickly and easily identify any changes by comparing them with what you find after the installation.

For changes to the WIN.INI and SYSTEM.INI files, make use of the WIN.INI and SYSTEM.INI Additions Editors built into NAIMAINT.EXE. An easy way to pull WIN.INI changes into a .DAT file is to follow this procedure:

1. Run NAIMAINT.EXE and select *CREATE NEW APPLICATION*.

2. At the Add Form, click on the *WIN.INI ADDITIONS*

VIEW/EDIT/LOAD Button to enter the WIN.INI Additions Editor.

3. Click on the *LOAD FROM FILE* Button, and specify the WIN.INI file from the configuration used to load the application to the file server.

4. Once the WIN.INI file is loaded, select only the text which was added during the installation of the application.
5. Click on the *DELETE ALL EXCEPT SELECTED* Button, and you will have in the editor only those changes applicable to the application you are adding. (The original WIN.INI file is not modified).

This same procedure will work equally well for any changes to the SYSTEM.INI file, using the *SYSTEM.INI ADDITIONS VIEW/EDIT/LOAD* Button from the Add Form.

Application Updates

When a new version of an application you are using is released, add the new version to WINAPPS.LST, but leave the old version in place as well. (You can distinguish the new version by adding a version number to the end of the List Name, and by using a different .DAT filename). Instruct your users to use N.A.I. to remove the old version and then to install the new version. After you are satisfied that everyone has upgraded, remove the old version from WINAPPS.LST and from the file server.

Initial End User Windows Configurations

You can configure the Windows setup program to automatically install the N.A.I. icon in the Main program group of every Windows installation. That way, you can let end users finish the job themselves by installing whatever Windows applications they need through N.A.I.

It is also very quick and easy to install local copies of Windows from a file server. Consult the Windows documentation for instructions on how to set Windows up to best allow this type of installation.

Three easy steps enable the Windows setup program to automatically install

N.A.I. to each new Windows installation:

1. Copy the file VBRUN100.DLL into the directory on the file server, or onto one of the diskettes, from which you will install Windows. If you are going to be installing Windows from diskettes, note the number of the diskette to which you have copied VBRUN100.DLL. If it does not fit onto any of the Windows diskettes, it will be simplest if you copy it manually into each Windows directory immediately after the Windows setup program completes.

2. In the same directory on the file server, (or on the first diskette, if you are installing Windows from diskettes) open the file SETUP.INF with the Windows Notepad or any other ordinary text editor.

   If you have copied the file VBRUN100.DLL to the Windows directory on the file server or to the first diskette, add the following line to the [win.apps] section:

   1:VBRUN100.DLL, "Visual Basic Runtime Module"

   The first character in this line (the number 1 in the example above) specifies on which Windows diskette the setup program can expect to find the VBRUN100.DLL file. If you are installing Windows from a file server, diskettes will not be involved, so you can just use the number 1. If you will be installing Windows on each network node from diskettes, this number should correspond to the number of the diskette to which you have copied the VBRUN100.DLL file. In either case, this line will assure that VBRUN100.DLL is automatically copied into the Windows directory of each new Windows installation. If you have been unable to copy VBRUN100.DLL to one of the Windows diskettes, do not add this line.

3. Finally, add to the [Main] section, the line which will create the N.A.I. icon in the Program Manager *Main* Group:

   "N.A.I.",F:\NETAPPS\NAI.EXE

   For the path *F:\NETAPPS\*, substitute the drive and directory where you have installed N.A.I. and the application .DAT files.

Installing Windows Applications to Run Locally

Heavy network use and a need for better performance may encourage you to load Windows applications on each PC's local hard disk. N.A.I. can make quick work of this type of installation, too, saving many hours of tedious support effort. The same procedures suggested for installing Windows applications on a network drive and running them from networked PCs will work for installing a separate copy of each application on each PC. The .DAT file will differ, of course, in what files are specified to be copied to the local disk, and in the path to the executable file and icon, but the principles remain the same. N.A.I. can enable all your end users to do the job themselves--quickly and easily, without the possibility of confusion.

REFERENCE NOTES:

Backup Files

WIN.INI

When a user runs N.A.I., the program makes a backup copy of the WIN.INI file, named WININI.TMP. When the program concludes, if it has changed the WIN.INI file, it renames WININI.TMP to WININI.BAK (first deleting any existing file named WININI.BAK). If it has not changed the WIN.INI file, it deletes WININI.TMP.

Application .DAT Files

If a new application .DAT file is created as a result of modifications to an existing .DAT file, the original .DAT file is preserved with a .BAK extension. (Again, if an earlier .BAK file exists, it is deleted to allow the current version to use that file name).

WINAPPS.LST

When the file WINAPPS.LST is updated, the previous version is retained with a .BAK extension. As with WIN.INI and the Application .DAT files, only the last version of WINAPPS.LST is preserved as WINAPPS.BAK.

File Formats

Formats for all data files associated with NAI.EXE and NAIMAINT.EXE are fully described here. All of these files are flat ASCII files, so they can be edited manually with the Windows Notepad, or any other ordinary text editor. However, the information provided here is intended for reference only; these files are created and maintained by the NAIMAINT.EXE program (except for NAI.INI, which is created and maintained automatically by the NAI.EXE program) and under normal circumstances should require no manual editing whatsoever.

WINAPPS.LST

```
;====================================================
;            FILE WINAPPS.LST
; WINDOWS APPLICATIONS LIST FOR N.A.I. rel. 1.5
;====================================================

; N.A.I. Remove capability:
REMOVE=ENABLED

; User-Specific Information:
$VAR1=Network Username
$VAR2=
$VAR3=
; Format for application list is:

; Application List Name|Application .DAT file name

Charisma|CHARISMA
CorelDraw|CORELDRW
Designer|DESIGNER
Excel|EXCEL
Instant Org Chart|INSTORG
Milestones|MILES3
Packrat|PACKRAT
Powerpnt|POWERPNT
Project for Windows|PROJECT
Word for Windows|WFW

;====================================================

; N.A.I. (Network Application Installer for Windows)

;       (c) Copyright 1991, 1992 by Aleph Systems
;     7319 Willow Avenue, Takoma Park, MD 20912
;                All rights reserved.

;          License: $99 (US) per file server

;====================================================
; End of file WINAPPS.LST
;====================================================
```

Notes:

Blank lines and lines beginning with ; are ignored.

Before the list of applications, the *REMOVE=* line must appear. The only recognized values for this option are *ENABLED* and *DISABLED*. It is this setting which determines whether or not the *REMOVE* Button appears in the N.A.I. program, enabling users to remove listed applications from their local configurations.

User-specific variables are indicated in the WINAPPS.LST file by the *$VAR1=*, *$VAR2=*, and *$VAR3=* lines. If the line is empty following the "=", then the variable is not in use. If a variable is in use, then the text which follows the "=" will be used by the NAI.EXE program to prompt the user for the specific information needed at installation time.

Each line represents one application, with the display name of the application (what the user sees in the list of available applications) first, followed by a vertical bar and then the filename (no path, no extension) of the data file containing the information needed to install the program.

Application .DAT Files

```
;=========================================================
; File DESIGNER.DAT (N.A.I. .DAT file for Designer)
;=========================================================
@WIN.INI
[Extensions]
drw=designer.exe ^.drw
shw=mgxslide.exe ^.shw

[Micrografx]
Designer 3.1=F:\DESIGNER
Libraries 4.0=F:\MGXLIBS
CLIPART=F:\MGXCLIP
Libraries=F:\MGXLIBS

[Clipboard Formats]
MGX_DRAW=0,"Micrografx Picture"
MGX_PICT=0,"In*a*Vision or Windows Draw Picture"

[Micrografx Outline Fonts]
BSBEZIER=F:\MGXLIBS\BITFONTS,MGXBITBZ,BSBEZ.FTM,0,15
BSSPEEDO=F:\MGXLIBS\SPDFONTS,MGXBITSP,BSSPD.FTM,0,20
URWBEZIER=F:\MGXLIBS\URWFONTS,MGXURWBZ,URWBEZ.FTM,0,6

[MGXTIFF]
Format=64
Device=1
Compression=2

[MGX_DRW_OUTPUT]
CPI=480

[CGMIN]
CPI=4052
TypeOfCGM=Harvard Graphics
ShowProfile=0

[Designer 3.1]
Warning=1
AutoPaste=0
Format=64
AutoPaste=0
```

```
@SYSTEM.INI

@FILES
F:\DESIGNER\DESIGNER.INI @WINDOWS

@ICONS
XGROUP=Main
F:\DESIGNER\DESIGNER.EXE,Designer,F:\DESIGNER\DESIGNER.EXE,,,,F:\$VAR1$\DATA

;=====================================================

; Network Application Installer for Windows rel. 1.5

;           (c) 1991, 1992  by Aleph Systems
;       7319 Willow Avenue, Takoma Park, MD 20912
;                 All rights reserved.

;           License: $99 (US) per file server

;=====================================================
; End of File DESIGNER.DAT
;=====================================================
```

Notes:

Blank lines and lines beginning with ; are ignored.
Three lines are mandatory in the .DAT file (a 4th, @SYSTEM.INI, will always be added when a .DAT file is created by NAIMAINT.EXE release 1.5--but to preserve compatibility with earlier releases of N.A.I., this line can be omitted), and they must appear in this order:

@WIN.INI
@SYSTEM.INI
@FILES
@ICONS

Any changes to the user's WIN.INI file must appear below the *@WIN.INI* heading, exactly as they are to appear in the user's WIN.INI file. If there are no changes to WIN.INI, the heading must still appear, but it would be followed on the next uncommented line by the *@SYSTEM.INI* heading.

Any changes to the user's SYSTEM.INI file must appear below the *@SYSTEM.INI* heading, exactly as they are to appear in the user's SYSTEM.INI file. If there are no changes to SYSTEM.INI, the heading should still appear, but it would be followed on the next uncommented line by the *@FILES* heading.

In order to preserve compatibility with application .DAT files created by earlier versions of N.A.I., release 1.5 has been constructed so that the @SYSTEM.INI heading is not required, but any .DAT files created with NAIMAINT.EXE release 1.5 will have this heading in them, whether or not any changes to SYSTEM.INI are required.

Any files which must be copied to the user's local hard disk must appear below the *@FILES* heading, in the following format:

sourcefile destinationdirectory

DOS wildcards (* and ?) are OK, but please see the discussion of **Wildcard Deletions** later in the **Reference Notes** section. The *sourcefile* must include drive and path. The *destinationdirectory* assumes the drive where Windows was started, unless another drive is specified in the path. Any directory in the *destinationdirectory* will automatically be created on the user's local drive, if necessary.

If the *destinationdirectory* is omitted, or if *@WINDOWS* appears in its position, the file(s) will be copied to the user's Windows directory (where his WIN.INI file is found--usually C:\WINDOWS). If *@SYSTEM* appears in the *destinationdirectory* position, then the file(s) will be copied to his Windows\System directory, wherever that may be (usually C:\WINDOWS\SYSTEM).

If you need to place a user-specific directory into the *destinationdirectory*, N.A.I. now provides full support for user-specific variables. Within the *destinationdirectory*, you would insert the appropriate $VAR code, and N.A.I. will substitute at runtime the corresponding user-specific information. See the User-Specific Variables section, above, for complete details.

If no files need to be copied, the heading must still appear, but it would be followed on the next uncommented line by the *@ICONS* heading.

Information for each icon to be created in the user's workspace must follow the *@ICONS* heading. If the icons are to go in a separate icon group, then the name of that group should appear in the first uncommented line beneath the *@ICONS* heading, in the following format:

GROUP=groupname

If the icons are to go into an existing icon group, then the name of that group should appear in the first uncommented line beneath the *@ICONS* heading, in the following format:

XGROUP=groupname

The *GROUP=* or XGROUP= line is optional. If it is omitted, N.A.I. will install the icon(s) in the icon group which is current (active) at runtime.

Each icon to be placed must have a line in this final section of the .DAT file. The format for the icon information is as follows:

executablefilename,displayname,iconpath,,,,workingdirectory

The *executablefilename* should include a full, specific path to the program file to be executed when the icon is double-clicked. $VAR codes are valid within the *executablefilename*.

The *displayname* is whatever should appear beneath the icon in the user's workspace.

The *iconpath* indicates a file from which to draw the icon; it may be different from the executable file. $VAR codes are valid within the *iconpath*. Only the comma should separate the items (no spaces or tabs).

The *workingdirectory* specifies the default data directory for the application (in version 3.1 and above of Windows--if the end user is running an earlier version of Windows, the *workingdirectory* information is ignored). $VAR codes are valid within the *workingdirectory*. (**NOTE:** the *workingdirectory* must be preceded by exactly 4 commas) .

NOTE: If any of these items are omitted, it is important that the commas appear anyway. Only the commas should separate the items (no spaces or tabs).

The group specified in the *GROUP=* line will be automatically created by N.A.I., and all icons specified will be created in this group. If a group is specified in the X*GROUP=* line instead of a *GROUP=* line, that group will be automatically created by N.A.I. if it does not already exist, and all icons specified will be created in this group. If no *GROUP=* or *XGROUP=* line appears, the icons will be created in the group which is current (active) in the user's Windows workspace at the time when the installation is performed. While it is technically not necessary to include any icon information (the *@ICONS* heading must still appear in the .DAT file, though), the user would have no way to run the program after installation if no icon was installed.

NAIMAINT.CFG

```
;=====================================================
; File NAIMAINT.CFG (NAIMAINT.EXE Config file)

; Comments begin with '

; Blank lines are ignored
;=====================================================
DATDIR=F:\NETAPPS

;=======================================================

; Network Application Installer for Windows rel. 1.5

;            (c) 1991, 1992 by Aleph Systems
;      7319 Willow Avenue, Takoma Park, MD 20912
;                  All rights reserved.

;           License: $99 (US) per file server

;=======================================================
; End of File NAIMAINT.CFG
;=======================================================
```

Notes:

Blank lines and lines beginning with ; are ignored.

The only valid configuration option is the location of the working directory, specified in the *DATDIR=* line. All other lines must either be blank or begin with ;.

NAI.INI

```
[Programs]
Designer 2.0=10:03:23,01/07/92
Excel 3.0=14:02:45,02/14/92
PowerPoint=14:03:15,02/14/92
Word for Windows 2.0=14:04:00,02/14/92
```

Notes:

Blank lines and lines beginning with ; are ignored.

This file is a standard format Windows application initialization file. In the current release, the only valid heading is *[Programs]*. Each line beneath that heading has to the left of the equal sign the Application List Name of the application installed by N.A.I. To the right of the equal sign is the time of installation, followed by a comma and the date of installation.

In the current release, this file is used only during the N.A.I. Install Application function, to determine whether or not an application has been installed previously. Later releases will employ this file for the purpose of automatically determining when an application upgrade is needed.

Icon Groups

If an application .DAT file specifies that a separate icon group be created, then the *REMOVE* operation of N.A.I. will automatically remove that group and all its icons from the user's workspace. If the .DAT file specifies installing the icons into an existing icon group or into the icon group which is current (active) at runtime, then the *REMOVE* operation will make no attempt to remove any icons. Instead, when the rest of the removal process is complete, N.A.I. will inform the user that the icon(s) may remain in the workspace and may have to be removed manually. Simple instructions for removing an icon are included in this message.

Note that during the installation of an application, the CREATE SEPARATE ICON GROUP and ADD TO EXISTING ICON GROUP work the same way. That is, if the icon group does not exist, N.A.I. will create it and install the icons in it. On the other hand, if the specified icon group *does* exist, then N.A.I. will simply add the icons to those already in the specified group.

The difference between these two choices arises only when N.A.I. removes the application. The *REMOVE* operation for any application specifying CREATE SEPARATE ICON GROUP will remove the entire icon group and all the icons it contains, whether they are part of the application being removed or not. In contrast, the *REMOVE* operation for an application specifying ADD TO EXISTING ICON GROUP will remove no icons or icon groups. Instead, it will behave exactly the same as the ADD TO CURRENT (ACTIVE) ICON GROUP option: it will inform the user that the icons may have to be removed manually, and it will at the same time provide some brief instructions on how to do so.

Reinstallation Check

When a user clicks the *INSTALL APPLICATION* Button in N.A.I., the program checks the user's local configuration to determine whether or not the selected application is already installed. Beginning with release 1.5, N.A.I. keeps in the user's Windows directory the file NAI.INI, a log of all applications it has installed and the date and time it installed them. If N.A.I. does not find an application listed as having been installed in the NAI.INI file--and does not encounter a match between the [Extensions] heading in the WIN.INI additions section of the application .DAT file and the user's WIN.INI file, it assumes that the application has not been installed. If N.A.I. believes that the application is currently installed, it will give the user the option of aborting the installation or reinstalling the application.

Because of idiosyncrasies in various Windows applications and the way that they share files and WIN.INI headings, N.A.I. assumes that an application is NOT currently installed unless it finds the application listed as having been installed in the NAI.INI file or it finds in the WIN.INI [Extensions] section the extensions specified in the application .DAT file. If neither of these conditions exists, N.A.I. will assume that the application is not installed.

If N.A.I. reinstalls an application (whether or not it is aware that it is reinstalling), the affected WIN.INI sections will be entirely rewritten according to the information in the .DAT file, and the files specified in the .DAT file will replace any by the same name which are already located on the user's local disk, and new icons will be added to the user's workspace.

Shared WIN.INI Headings

Certain applications share WIN.INI headings with other applications. For example, a number of Microsoft applications use the [Microsoft Help] heading in WIN.INI, while a number of applications from Micrografx, Inc. share the [Micrografx] heading. If a user has installed in his configuration two applications which share a heading, this heading may cause temporary problems if N.A.I. is used to remove one of the two applications.

In such a case, the remaining application will find that the heading it needs is missing from WIN.INI. If you suspect that N.A.I. has removed a shared WIN.INI heading, try reinstalling the remaining application. The reinstallation will replace the missing heading and the remaining application should then work as before.

Special WIN.INI Headings

For most WIN.INI headings, N.A.I. treats the entire heading and all the lines beneath it as a single unit, installing it or removing it whole cloth. However, N.A.I. treats three WIN.INI headings in a special fashion. The [Extensions], [Embedding], and [OLE] headings are all known to be shared. Therefore, N.A.I. inserts or deletes from these headings only the information which the application .DAT file specifies belongs under them. The result is that for these three headings, each application .DAT file will affect the only the lines it specifies; all other lines beneath these headings will remain untouched.

SYSTEM.INI Headings

For SYSTEM.INI headings, N.A.I. inserts or deletes only the information which the application .DAT file specifies belongs under them. The result is that for the SYSTEM.INI file, each application .DAT file will affect the only the individual lines it specifies; all other lines will remain untouched.

Wild Card Deletions

N.A.I. supports the use of DOS wildcards (* and ?) in the specification of files to be copied. However, this support has important safety implications for N.A.I.'s application removal feature. N.A.I. includes special safeguards to prevent the application removal feature from inadvertently causing damage to a user's Windows configuration or other areas of his local PC. Nevertheless, the application removal feature is very powerful, and it is important for a LAN administrator to understand how it operates in order to be certain that it works as intended. Remember that the application removal feature can be disabled from within the NAIMAINT.EXE program.

If a user clicks on the *REMOVE APPLICATION* Button in N.A.I., the program will attempt to remove from the local configuration whatever files the application .DAT file specifies as required to be copied to the local configuration during installation of the application. If the source files are specified by complete filenames, they are simply deleted one by one, by complete filename. This method is completely safe. However, if these files are specified by means of wildcards, N.A.I. follows a cautious procedure for their removal, to avoid removing any files other than those intended:

1. N.A.I. first looks for the individual filenames in the specified source directory on the file server. If files matching the file specification are found in the source directory, then N.A.I. uses these names in the removal process: it deletes each of these filenames one by one from the user's local configuration. In other words, if the .DAT file specifies F:\FONTS\*.FNT as the source file and C:\FONTS\ as the destination directory, N.A.I. will look in the F:\FONTS\ directory for files with the .FNT extension. If it finds, say, FONT1.FNT, FONT2.FNT, and FONT3.FNT, then it will perform three separate deletions on the local drive, equivalent to the following series of DOS commands: DEL C:\FONT1.FNT, DEL C:\FONT2.FNT, and DEL C:\FONT3.FNT.

2. With certain exceptions for reasons of safety (see #3, below), if the file server does not contain any files matching the source file specification in the .DAT file, then N.A.I. performs a wild card deletion of files matching the specification on the local configuration. For example, if the source file specification reads F:\FONTS\*.FNT but the F: drive holds no files with the .FNT extension in the \FONTS\ directory (perhaps because the application has been removed from the file server), then N.A.I. will perform on the local drive the equivalent of the following DOS command: DEL \FONTS*.FNT.

3. *If the destination directory is the WINDOWS or SYSTEM directory or the ROOT directory of any drive, then N.A.I. will not delete files specified in the .DAT file by the use of wildcards unless it can discover the individual filenames from the file server,* as discussed in #1, above. In other words, **if** the .DAT file specifies files by the wildcard method **and** the source files are not found in the source directory **and** the destination for those files is the WINDOWS directory, or the SYSTEM directory, or a ROOT directory, **then** N.A.I. will **not** delete the files.

LICENSE AGREEMENT

Disclaimer

This program is provided without any express or implied warranties whatsoever. Because of the diversity of conditions and hardware under which this program may be used, no warranty of fitness for a particular purpose is offered. The user is advised to test the program thoroughly before relying on it. The user must assume the entire risk of using the program. The manufacturer assumes no liability of any kind.

Licensing Information

This copyrighted program is NOT free. It is shareware: you are permitted to ***test*** it in your network to see if it will prove useful to you. ***If you do actually put it into use***, you are legally obligated to license your copy from the manufacturer, Aleph Systems. You are encouraged to pass a copy of this program along to others for evaluation, but please include all of the files, and please encourage others to license their copies as well. The program, its logic and ease of use all represent the fruits of many hours of labor. If you use this program, you are both legally and ethically bound to pay the small license fee. Please do not assume that the fees paid by other users will somehow lessen your obligation. Even more importantly, do not assume that the program's low price reflects low value. This program can save you enormous amounts of time, effort, and money: if you use N.A.I. to install even one application on most of the nodes in an average sized network, you will have saved enough time to easily cover the expense of a license.

An invoice/order form is provided for your convenience at the end of this document.

To keep licensing as simple and equitable as possible, this program is licensed on a per-file server basis, at $99 per server. For the purposes of this license, a server is defined as any computer from which the program is made available to be run. In other words, even if you are not running N.A.I. from a dedicated file server (for example, in a peer-to-peer network which has no dedicated file servers), any computer from which the N.A.I. program is made available is considered a file server and requires a $99 license. You may elect to run multiple copies of this program on a single server; in this case, the cost to you is still only $99. However, if you install a copy of this program on a second file server, you are obligated to license the copy on that second file server at a cost of an additional $99.

For pricing information on licenses for 10 servers or more or on upgrades from previous releases, please contact Aleph Systems directly, either by mail at 7319 Willow Avenue, Takoma Park, MD 20912, or by CompuServe mail at 71371,635.

Anyone distributing this program for any kind of remuneration must first contact Aleph Systems for authorization.

INVOICE

Network Application Installer for Windows

Aleph Systems
7319 Willow Avenue
Takoma Park, MD 20912

Company Name: ____

Address: ______________________ Network Operating System and Version:

______________________ ______________________

______________________ File Server Make and Model:

______________________ ______________________

Point of Contact: ______________________ Phone: ______________

_______ Licenses @ $99 per file server: ______________

+ Shipping & Handling: $10 ______________
(Optional---includes printed documentation and latest version of N.A.I.):

+ 5% Sales Tax (MD only): ______________

TOTAL ENCLOSED: ______________
(Please make check payable to Aleph Systems)

Comments, suggestions, problems:

____