VisualDOC

INSTALLATION GUIDE

VERSION 6.2

AUG 2009

1767 SOUTH 8TH STREET, SUITE 200
COLORADO SPRINGS, CO 80905
Phone: (719) 473-4611 Fax: (719) 473-4638
http://www.vrand.com
email: visualdoc.support@vrand.com

1 General Installation

1.1 Introduction

This document provides a detailed description for installing and running all VisualDOC and DOT software components provided by Vanderplaats Research and Development, Inc. (VR&D). It is not required to read the entire document. However, it is recommended to carefully read this section before installing any software components. The remaining sections are optional and self-contained.

The VisualDOC CD-ROM from VR&D contains a number of general purpose optimization software components, as summarized in the table below. Most users will install and use only VisualDOC itself. VisualDOC is VR&D's general-purpose optimization software system which includes VisualScript. VisualScript is a graphical tool that allows the user to couple external analysis programs with VisualDOC.

Software Component	Description
VisualDOC/VisualScript	General-purpose Optimization Software System
VisualDOC C/C++ API	VisualDOC Application Programming Interface
DOT	Gradient-based Optimization Library
BIGDOT	Large Scale Gradient-based Optimization Library
FLEXlm License Server	FLEXlm License Server Tools

The VisualDOC C/C++ API is an application programming interface that allows users to embed the VisualDOC technology inside their own analysis programs. DOT and BIGDOT are optimization libraries (licensed separately) for gradient-based optimization that users can embed into their own analysis programs. The DOT and BIGDOT libraries provide only a subset of the VisualDOC C/C++ API functionality, specifically targeted to gradient-based optimization. BIGDOT is targeted to large scale optimization problems with large number of design variables.

The FLEXlm license server provides VR&D customers with a floating network license capability, allowing users to setup a single license server machine that serves licenses to client machines connected by a local area network. The products and number of simultaneous licenses that are served depend on the floating licenses purchased from VR&D. A license server running on one platform can serve licenses to clients running on any other supported platform.

Note: Windows 98 machines cannot be used as license servers. However, these machines can be used as clients that interact with a license server to obtain licenses for running VR&D software.

1.2 Supported Platforms

All VisualDOC software components are supported on a wide range of platforms as summarized below:

Platform	Operating System
Sun	Solaris 9 and higher
PC	Windows 98/NT/2000/XP
Linux32	Linux 32 Bit, Kernel 2.6 and higher
Linux64	Linux 64 Bit, Kernel 2.6 and higher

Please feel free to contact VR&D if any VisualDOC software component(s) are needed for a different platform.

1.3 Installing the Software

Default Installation

Installing the software is generally straightforward. By default only VisualDOC itself, along with associated documentation and example problems, is installed. The default installation should meet the needs of most users.

On UNIX machines (including Linux and AIX), simply run the *install.sh* script in the root directory of the VisualDOC CD-ROM. For a detailed discussion on how to mount the CD-ROM on any of the supported UNIX platforms, please see Appendix A.

On Windows machines, the installation script should start automatically when the CD-ROM is inserted. If the autorun feature is turned off, simply run the *setup.exe* program in the *pc* sub-directory of the VisualDOC CD-ROM.

Installing Additional Components

In addition to the default installation, the user may also choose to optionally install the VisualDOC C/C++ API, the DOT/BIGDOT libraries, or the FLEXlm license server.

Note: The VisualDOC C/C++ API requires a valid VisualDOC installation.

To install additional components, simply select the desired component(s) from the list of available components provided by the installation script.

1.4 Additional Required Software

A valid Java Runtime Environment (JRE) is required to run VisualDOC and VisualScript. None of the other VisualDOC and DOT software components require any additional software. VisualDOC requires at least JRE 1.5.0, but it is highly recommended to use the latest available version. The latest available JRE versions at the time this document was created are summarized below on a per platform basis.

Operating System	JRE Version
Solaris	JRE 1.6.0_07
Windows	JRE 1.6.0_07
Linux 32 Bit	JRE 1.6.0_07
Linux 64 Bit	JRE 1.6.0_07

The Windows JRE is provided in the *pc* directory. However, the Windows installation will automatically ask the user to install the JRE during the VisualDOC installation. For UNIX platforms, the JRE are available for download, and must be installed before installing VisualDOC. Only the Java Runtime Environment (JRE) and not the Software Development Kit (SDK) is required to run VisualDOC. The JRE can be downloaded, free of charge, from the respective hardware vendors, using the following web sites:

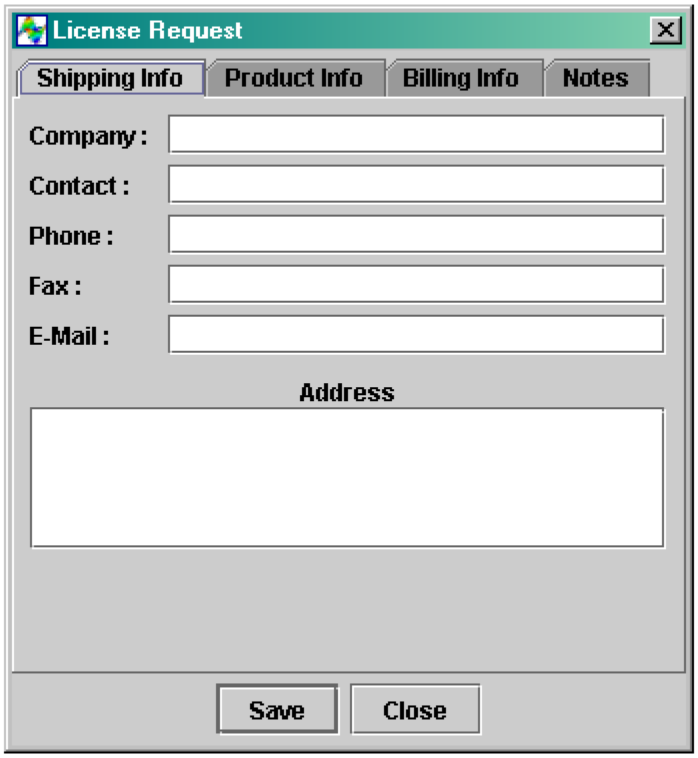
Operating System	JRE Version
Solaris	http://www.java.com/en/download/manual.jsp
Windows	http://www.java.com/en/download/manual.jsp
Linux 32 Bit	http://www.java.com/en/download/manual.jsp
Linux 64 Bit	http://www.java.com/en/download/manual.jsp

1.5 Obtaining a VR&D License File

All VisualDOC software components require a license to run. Without a valid license, all software components will revert to demo mode which will result in limitations with regard to the problems that can be solved. To evaluate a full version of any software component, contact VR&D for a free evaluation license.

Once the installation is finished, a license request must be completed and e-mailed to VR&D. To complete a license request, run the provided *request* program as shown in **Figure 1-1**. To obtain the corresponding license file, select the **Save** button and e-mail the resulting request file to VR&D. On UNIX machines the *request* program is located in the *<install_dir>/vrand/bin* directory. On Windows machines the *request* program is located in the file *<install_dir>/vrand/vdoc6.2/WIN32* directory, but can also be accessed from the Windows **Start** menu. Alternatively, the file *request* program may be launched from within the VisualDOC or VisualScript GUI by selecting the **License Request...** option from the **Help** menu.

VisualScript, the VisualDOC optimization libraries, the VisualDOC C/C++ API, and DOT & BIGDOT are licensed separately. Additionally, the user can request either a node-locked license or a floating network license.



Company :	
Contact :	
Phone :	
Fax :	
E-Mail :	

Address

Figure 1-1VisualDOC license request program

If only DOT/BIGDOT is installed, the *dot_request* program can be used to request a license. The *dot_request* program is provided in the *<install_dir>/vrand/dot5.7* directory on UNIX machines and the *<install_dir>/vrand/dot5.7/WIN32* directory on Windows machines. Alternatively, the *dot_request* program can also be accessed from the Windows **Start** menu. The *dot_request* program will create a request file that should be e-mailed to VR&D to obtain the corresponding license file. If only the FLEXlm license server is installed, the *lmutil* program can be used to obtain a **lmhostid**. The **lmhostid** can be e-mailed to VR&D with a description of the request.

Although several options for requesting a VR&D license file are provided, the first option of using the VisualDOC *request* program is preferred and should always be used if VisualDOC is installed.

1.6 Potential Problems

Potential Problems on Windows

Installation does not complete

The VisualDOC installation program uses a recent version of the common controls DLL *Comctl32.dll* on Windows. Some older Windows computers may not have this recent version of the common controls DLL installed. This may be the case when an old version of Internet Explorer (4.0 or older) exists on a computer. In these cases, the

VisualDOC installation process will not complete and the user will be asked to upgrade the computer.

TO FIX THIS PROBLEM run the *50comupd.exe* file that is provided in the *pc* directory of the VR&D CD-ROM. Running this file will install the latest version of *Comctl32.dll*. Restart the computer for the new changes to take effect. Repeat the VisualDOC installation process after restarting the computer.

HOME directory

VisualDOC uses the *HOME* directory of each user to save user specific information. Unfortunately, some computers may not have a valid *HOME* directory set for each user. In this case VisualDOC will complain that it can not find the *HOME* directory during startup.

TO FIX THIS PROBLEM set the *VHOME* environment variable to point to a directory that will be considered the *HOME* directory for the current user when running VisualDOC. See Appendix B for a description of how to set/edit environment variables on Windows computers.

Missing *dformd.dll* library

Both the VisualDOC C/C++ API and DOT/BIGDOT libraries require the *dformd.dll* dynamically linked FORTRAN runtime library. This library is automatically installed with both the VisualDOC C/C++ API and/or DOT/BIGDOT installations. However, the system path may have to be modified to ensure that an application linked with the VisualDOC API or the DOT/BIGDOT libraries can find the *dformd.dll* library

TO FIX THIS PROBLEM, set the system *PATH* environment variable to point to the *<install_dir>/vrand/vdoc6.2/WIN32* directory (or the *<install_dir>/vrand/dot5.7/WIN32* directory if only DOT/BIGDOT is installed) before running any application that is linked with the API or DOT/BIGDOT libraries. See Appendix B for a description of how to set/edit environment variables on Windows computers.

Repaint problems and/or random crashes

Some Windows 98 computers are not set up to handle Java-based applications very well. VisualDOC is a Java-based application. Because of this combination, VisualDOC may experience repaint problems and/or random crashes when running on Windows 98 platforms.

TO FIX THIS PROBLEM, turn down the display hardware acceleration. To turn down the hardware acceleration, go to the **Start** menu and select the **Settings/Control Panel/System** option. This will bring up the **System Properties** dialog box. Select the **Performance** tab. In the **Performance** tab select the **Graphics** button to open the **Advanced Graphics Options** dialog. In this dialog box, change the hardware acceleration to **None**.

2 VisualDOC and VisualScript

2.1 Introduction

VisualDOC is VR&D's general-purpose optimization software system and is generally used to wrap optimization around existing analysis programs. VisualDOC includes VisualScript, which is a graphical tool for coupling existing analysis programs with VisualDOC.

VisualDOC and VisualScript come with on-line context documentation that can be accessed by selecting the **Help**... option from the **Help** menu. A more detailed documentation is also provided in the form of *.pdf* documents in the *<install_dir>/vrand/vdoc6.2/docs/pdf* directory. These include the following:

File	Description
DownloadInstructions.pdf	Download Instructions
WhatsNew.pdf	New Features in this Release
InstallationGuide.pdf	This Document
UsersManual.pdf	VisualDOC User's Manual
GettingStartedExamples.pdf	VisualDOC Getting Started Manual
AdvancedExamples.pdf	VisualDOC Advanced Examples Manual
TheoryManual.pdf	Theory Manual on Algorithms included in VisualDOC
C_API.pdf	API Manual for Accessing VisualDOC Functionality from inside a Program

2.2 Using VisualDOC and VisualScript on UNIX

The following scripts are provided in both the *<install_dir>/vrand/*bin and *<install_dir>/vrand/*vdoc6.2/*$ARCH* directories (where *$ARCH* is one of: SunOS, Linux_ix86, Linux_x64) to allow the user to run different components of VisualDOC.

File	Description
visualdoc	VisualDOC GUI startup script
vscript	VisualScript startup script
request	VisualDOC license request program
runtask	VisualDOC run task command line utility

dbreport	Report command line utility
lmutil	FLEXlm license server utility

The command line utilities are provided to operate on existing VisualDOC databases. Before using any of the command line utilities, the appropriate VisualDOC tasks should be created using the VisualDOC GUI. For example *runtask* script can be used to run any existing VisualDOC task, created with the VisualDOC GUI, as follows:

```
runtask -d "house" -t 1
```

The above command will execute task number 1 in the *house* database. All command line arguments for any of the command line utilities can be obtained by simply running the appropriate utility without any options. For example, submitting the following command

```
runtask
```

will produce the command line options for *runtask* as follows:

```
runtask v6.20 (0) - General program to run VisualDOC tasks

Usage:
runtask -d "database name" -t # [-h] [-?] [-D #] [-N "log file name"] [-ef "exclude
file name"] [-b]
 -d "database name" : specifies the design database (required argument). The
database name must be surrounded by double quotes.
 -t # : # specifies the task number in the database to run (required argument).
 -h : displays this usage statement.
 -? : displays this usage statement.
 -D # : # specifies the amount of trace data to output to the log file. 0 - none, 1 -
intermediate, 2 - maximum
 -N "log file name" : By default the log file name is "vdoc_XXX.log", where
"XXX" is the number from 000 to 999 to make the name unique.
 -m : dump all available points for analyses at once.
 -p : perform parallel analysis (requires special setup).
 -np # : Perform parallel analysis on the provided number of processors (#) on a
SINGLE computer.  (# should be > 0).
 -ef "excluded file name" : specifies the excluded file (if multiple, separate with
';') for file copy during parallel computation.
 -b : Keep the intermediate input and output files (Parallel Computation).
 -w : if database is locked by somebody - wait for it to become unlock before
    starting running a task (if no flag - not to wait and exit).
```

2.3 Using VisualDOC and VisualScript on Windows

On Windows platforms the VisualDOC installation creates shortcut icons for VisualDOC and VisualScript on the computer Desktop. To run any of these two programs, simply select the appropriate icon from the Windows Desktop.

Alternatively, select **VisualDOC** or **VisualScript** from the **Start/Programs/vrand** menu.

To run the VisualDOC command line utilities, the following executable are provided in the *<install_dir>/vrand/vdoc6.2/WIN32* directory:

File	Description
`visualdoc.exe`	`VisualDOC GUI startup program`
`vscript.exe`	`VisualScript startup program`
`request.exe`	`VisualDOC license request program`
`runtask.exe`	`VisualDOC run task command line utility`
`dbreport.exe`	`Report command line utility`
`lmutil.exe`	`FLEXlm license server utility`

The command line utilities are provided to operate on existing VisualDOC databases. Before using any of the command line utilities, the appropriate VisualDOC tasks should be created using the VisualDOC GUI. For example *runtask* script can be used to run any existing VisualDOC task, created with the VisualDOC GUI, as follows:

```
runtask -d "piston" -t 1
```

The above command will execute task number 1 in the *piston* database. All command line arguments for any of the command line utilities can be obtained by simply running the appropriate utility without any options. For example, submitting the following command

```
runtask
```

will produce the command line options for *runtask* as follows:

```
runtask v6.20  (0) - General program to run VisualDOC tasks

Usage:
runtask -d "database name" -t # [-h] [-?] [-D #] [-N "log file name"] [-ef "exclude
file name"] [-b]
 -d "database name" : specifies the design database (required argument). The
database name must be surrounded by double quotes.
 -t # : # specifies the task number in the database to run (required argument).
 -h : displays this usage statement.
 -? : displays this usage statement.
 -D # : # specifies the amount of trace data to output to the log file. 0 - none, 1 -
intermediate, 2 - maximum
 -N "log file name" : By default the log file name is "vdoc_XXX.log", where
"XXX" is the number from 000 to 999 to make the name unique.
 -m : dump all available points for analyses at once.
 -p : perform parallel analysis (requires special setup).
 -np # : Perform parallel analysis on the provided number of processors (#) on a
SINGLE computer.  (# should be > 0).
 -ef "excluded file name" : specifies the excluded file (if multiple, separate with
';') for file copy during parallel computation.
 -b : Keep the intermediate input and output files (Parallel Computation).
```

-w : if database is locked by somebody - wait for it to become unlock before starting running a task (if no flag - not to wait and exit).

Note: The command line utilities should only be run from the command prompt. Trying to run a command line utility from the file explorer, by double clicking on it will result in a command prompt window that pop up and disappear immediately.

To allow the execution of the command line utilities in any directory without always providing the full path, it is recommended to add the *<install_dir>/vrand/vdoc6.2/WIN32* directory to the *PATH* environment variable.

See Appendix B for a description of how to set/edit environment variables on Windows computers.

3 VisualDOC C/C++ Application Programming Interface

3.1 Introduction

While VisualDOC is generally used to wrap optimization around existing analysis programs, the VisualDOC C/C++ API is provided as an additional tool that allows users to embed the VisualDOC functionalities inside their own analysis programs. Although the API is a very powerful tool, it does require programming from the user and is intended for advanced users only.

The API calls should be put into the user's analysis programs and the resulting software should be linked with the provided API library.

A detailed users manual for the VisualDOC C/C++ API is provided as *C_API.pdf* in the *<install_dir>/vrand/vdoc6.2/docs/pdf* directory.

3.2 Using the VisualDOC C/C++ API on UNIX

Using the VisualDOC C/C++ API requires users to programmatically implement the API calls in their own analysis programs and link the resulting software with the provided API library. On UNIX machines, the API library is provided as a shared library named *libVDOC_C_API.so*, installed in the *<install_dir>/vrand/vdoc6.2/$ARCH* directory (where *$ARCH* is one of SunOS, Linux_ix86, Linux_x64).

The following cases illustrate how to setup the environment and compile the VisualDOC C/C++ API with a user's code that implement the API. In these examples, **<outname>** is the name of the resulting executable program and **<example.c>** is the user's C code that implements the API calls.

Note: Machine specific makefiles are provided with the API example problems in the API example problem directory.

Solaris

For Solaris systems, the *csh* shell and the SUN C Workshop Compilers Version 5.4 are used in the example below, however neither of these are required.

The first step is to setup two environment variables that point to the authorization file and the API shared library respectively, as follows:

```
setenv VRAND_AUT <install_dir>/vrand/licenses/vdoc.lic
setenv LD_LIBRARY_PATH <install_dir>/vrand/vdoc6.2/SunOS/:\
<install_dir>/vrand/vdoc6.2/SunOS/stubs/:\
<install_dir>/vrand/vdoc6.2/SunOS/python/lib/
```

Next, invoke the compiler using the following command:

```
cc -o <outname> <example.c> -I<install_dir>/vrand/vdoc6.2/include\
-L <install_dir>/vrand/vdoc6.2/SunOS -lVDOC_C_API\
-L <install_dir>/vrand/vdoc6.2/SunOS/stubs -lmx -leng -lnsl -lsocket
```

Linux 32 Bit

For Linux systems, the *bash* shell and the gcc 3.3.3 compiler are used in the example below. The *bash* shell is not required, however, due to backwards compatibility problems, a gcc 3.3.3 or newer compiler may be required.

The first step is to setup two environment variables that point to the authorization file and the API shared library respectively, as follows:

```
export VRAND_AUT=<install_dir>/vrand/licenses/vdoc.lic
export LD_LIBRARY_PATH=<install_dir>/vrand/vdoc6.2/Linux_ix86/:\
<install_dir>/vrand/vdoc6.2/Linux_ix86/stubs/:\
<install_dir>/vrand/vdoc6.2/Linux_ix86/python/lib/
```

Next, invoke the compiler using the following command:

```
gcc -O2 -o <outname> <example.c> \
-I <install_dir>/vrand/vdoc6.2/include/ \
-L <install_dir>/vrand/vdoc6.2/Linux_ix86/ -lVDOC_C_API
```

Linux 64 Bit

For 64 bit Linux systems, the *bash* shell and the gcc 3.4.4 compiler are used in the example below. The *bash* shell is not required, however, due to backwards compatibility problems, a gcc 3.4.4 or newer compiler may be required.

The first step is to setup two environment variables that point to the authorization file and the API shared library respectively, as follows:

```
export VRAND_AUT=<install_dir>/vrand/licenses/vdoc.lic
export LD_LIBRARY_PATH=<install_dir>/vrand/vdoc6.2/Linux_x64/:\
<install_dir>/vrand/vdoc6.2/Linux_x64/stubs/:\
<install_dir>/vrand/vdoc6.2/Linux_x64/python/lib/
```

Next, invoke the compiler using the following command:

```
gcc -O2 -o <outname> <example.c> \
-I <install_dir>/vrand/vdoc6.2/include/ \
-L <install_dir>/vrand/vdoc6.2/Linux_x64/ -lVDOC_C_API
```

3.3 Using the VisualDOC C/C++ API on Windows

Using the VisualDOC C/C++ API requires users to programmatically implement the API calls in their own analysis programs and link the resulting software with the provided API library. On Windows machines the API is provided as a dynamically linked library *VDOC_C_API.dll* with corresponding stub library (the stub library is required when linking) *VDOC_C_API.lib*. These libraries were created using Microsoft Visual C++ 6.0 Developer Studio, but should work with any compiler that supports the use of DLL's. The dynamically linked library is located in the *<install_dir>/vrand/vdoc6.2/WIN32* directory. The corresponding stub library is located in the *<install_dir>/vrand/vdoc6.2/WIN32 /lib* directory.

The remainder of this section describes how to setup the environment and compile the VisualDOC C/C++ API with the user's code that implement the API calls. The first step is to set up the VRAND_AUT environment variable, pointing to the VisualDOC license file. The VisualDOC license file is typically named *vdoc.lic* and is located in the *<install_dir>/licenses* directory. See Appendix B for a description of how to set/edit environment variables on Windows computers.

Note: Sample Microsoft Visual C++ Version 6.0 project files are provided with the API example problems in the API example problem directory.

With the VRAND_AUT environment variable set, the next step is to make sure the C/C++ compiler (this discussion is for Microsoft Visual C++ Version 6.0) knows how to link with the VisualDOC API. The first step is to include the supplied stub library with the user's source files in the Visual Studio project used to build the user's program. The VisualDOC API dynamically linked library is only used at runtime and is not required during the linking stage. When executing the resulting program, the user should make sure that the program will be able to find the VisualDOC API dynamically linked library by modifying the PATH environment variable to include the directory that includes this library. The API dynamically linked library is installed in the *<install_dir>/vrand/vdoc6.2/WIN32* directory of the VisualDOC installation. See Appendix B for a description of how to set/edit environment variables on Windows computers.

The VisualDOC C/C++ API is compiled as a multithreaded dynamically linked library. To avoid potential problems when linking with the VisualDOC API, the user should make sure that their code is also compiled and linked as multithreaded dynamically linked code. In the Microsoft Visual Developer Studio, one has to select the **Project/Settings**... option from the main menu to make sure the code is compiled and linked with the correct options. The **Project Settings** dialog box will appear where the **C/C++** tab must be selected. From the **Category** drop-down list, select the **Code Generation** option and make sure the **Use run-time library** shows **Multithreaded DLL**.

4 DOT and BIGDOT Libraries

4.1 Introduction

The DOT and BIGDOT optimization libraries are similar to the VisualDOC C/C++ API in that it allows users to embed optimization directly within their own analysis program. However, DOT and BIGDOT provide only a subset of the VisualDOC C/C++ API functionality and is specifically targeted to gradient-based optimization. BIGDOT is specifically targeted to large scale optimization problems with large numbers of design variables. The DOT and BIGDOT libraries are not based on a central database and do not perform discrete optimization (BIGDOT can solve large scale discrete optimization problems), multi-objective optimization, response surface approximate optimization, or design of experiments.

DOT and BIGDOT are bundled in a single library that is installed in the *<install_dir>/vrand/dot5.7/$ARCH* directory (where *$ARCH* is one of SunOS, Linux_ix86, Linux_x64) and is licensed separately from VisualDOC. The installation includes both single and double precision libraries, example problems and documentation. Both the single and double precision libraries are provided as static and shared libraries. The supplied example problems include both DOT and BIGDOT examples and illustrate how to call DOT and BIGDOT from both FORTRAN and C.

Note: The example problems are supplied with machine specific makefiles. On the Windows platform, Microsoft Visual Studio Version 6.0 project files are provided.

Detailed DOT (*dot.pdf*), and BIGDOT (*bigdot.pdf*) user manuals are provided in the

<install_dir>/vrand/dot5.7/docs/pdf directory.

4.2 Using DOT and BIGDOT on UNIX

Before using the DOT/BIGDOT library it is necessary to set an environment variable that would allow the resulting executable to find the DOT or BIGDOT license file. For a typical installation in the *<install_dir>* directory, this variable should be set as follows for *csh* shell environments

```
setenv VRAND_AUT <install_dir>/vrand/licenses/vdoc.lic
```

and as follows for **bash** shell environments

```
export VRAND_AUT=<install_dir>/vrand/licenses/vdoc.lic
```

In the above examples, *<install_dir>/vrand/licenses/vdoc.lic* is the full path to the VR&D license file.

On UNIX machines, the DOT/BIGDOT library is supplied as both a static and a shared library as follows:

File	Description
libdot.a	Single precision DOT and BIGDOT static library
libDOT2.a	Double precision DOT and BIGDOT static library
libDOT.so	Single precision DOT and BIGDOT shared library
libDOT2.so	Double precision DOT and BIGDOT shared library

These library files are installed in the *<install_dir>*vrand/dot5.7/*$ARCH* directory (where *$ARCH* is one of SunOS, Linux_ix86, Linux_x64).

To use the DOT/BIGDOT library, simply invoke the FORTRAN compiler. For example; to use the single precision, static library on a Linux workstation one would typically use the following command

```
g77 -o <outname> <example.f> -L<install_dir>/vrand/dot5.7/Linux_ix86 -lDOT
```

where *<outname>* is the name of the resulting executable, *<example.f>* is the name of the source file that calls DOT and *-lDOT* is the name of DOT object library.

When using the DOT libraries on a Linux machine, it may be necessary to include the directory containing the DOT libraries in the shared library path. This is to ensure that the system finds the *libg2c.so.0* library, for example:

```
export LD_LIBRARY_PATH=<install_dir>/vrand/dot5.7/Linux\_x64
```

If the *libg2c.so.0* library is already available on your system this step is not required.

When using the DOT libraries on a SUN machine, it may be necessary to include the *socket*, *nsl*, and *dl* system libraries during the linking stage, for example:

```
f77 -o <outname> <example.f> \
-L<install_dir>/vrand/dot5.7/SunOS -lDOT -lsocket -lnsl -ldl
```

When using the shared libraries, it is required to set the shared library path to include the <*install_dir*>vrand/dot5.7/*$ARCH* directory before running the application. For example, to set the shared library path in a *csh* shell environment, use the following command

```
setenv <SHARE_LIB_PATH> <install_dir>/vrand/dot5.7/SunOS
```

where <*SHARE_LIB_PATH*> is platform specific, as follows

Platform	**SHARE_LIB_PATH**
SUN	LD_LIBRARY_PATH
Linux (32 Bit)	LD_LIBRARY_PATH
Linux (64 Bit)	LD_LIBRARY_PATH

4.3 Using DOT and BIGDOT on Windows

On Windows machines, the DOT/BIGDOT library is supplied as both static and dynamically linked libraries, with corresponding stub libraries, as follows:

File	**Description**
DOT_LIB.lib	Single precision DOT and BIGDOT static library
DOT2_LIB.lib	Double precision DOT and BIGDOT static library
DOT_LIB.dll	Single precision DOT and BIGDOT shared library
DOT2_LIB.dll	Double precision DOT and BIGDOT shared library
DOT_DLL.lib	Single precision stub library
DOT2_DLL.lib	Double precision stub library

These libraries are installed in the <install_dir>/vrand/dot5.7/*WIN32* directory.

Before using DOT or BIGDOT it is necessary to set the VRAND_AUT environment variable to point to VR&D license file. For a typical installation this variable should be set to point to <install_dir>*/licenses/vdoc.lic*. See Appendix B for a description of how to set/edit environment variables on Windows computers.

After the VRAND_AUT environment variable is set, make sure the FORTRAN compiler (Compaq Visual FORTRAN Version 6.0 is recommended) links with the correct DOT/BIGDOT library. When linking against a static library, simply include that library in the workspace. When linking against a dynamically linked library, include the corresponding stub library in the workspace. Sample project files are provided.

When using the DOT/BIGDOT dynamically linked library, remember that the dynamically linked library will be invoked only at runtime and is not required when linking the program. The resulting executable must be able to resolve the DOT/BIGDOT dynamically linked library at runtime. To ensure that the DOT/BIGDOT dynamically linked library is resolved, modify the *PATH* environment variable to include the directory where the dynamically linked library is located. The DOT/BIGDOT DLL is installed in the <install_dir>/vrand/dot5.7/*WIN32* directory. See Appendix B for a description of how to set/edit environment variables on Windows computers.

A final note on using the DOT dynamically linked library, all VR&D dynamically linked libraries are created as multi-threaded DLL's and the corresponding project files should be setup accordingly.

5 FLEXlm License Server

5.1 Introduction

VR&D make use of FLEXlm Version 10.8.0.1 to provide our customers with a floating network license capability. The FLEXlm license server provides VR&D customers with the ability to setup a single license server machine that serves licenses to client machines connected by a local area network. The products and number of simultaneous licenses that are served depend on the floating licenses purchased from VR&D.

The FLEXlm license server is available on all supported platforms, with one exception. Windows 98 machines cannot be used as license servers. However, these machines can be used as clients that interact with a license server to obtain licenses for running VR&D software. A license server running on one platform can serve licenses to clients running on any other supported platform.

The FLEXlm license server has to be installed and licensed separately. The FLEXlm license server will be installed into the <install_dir>/vrand/*flexlm* directory with sub-directories for documentation and platform specific programs. The platform specific programs are saved in the *$ARCH* sub-directory, where *$ARCH* depends on the machine architecture as follows: SunOS, AIX, Linux_ix86, Linux_x64, or WIN32. This platform specific directory contains all executable files required to setup and run the FLEXlm license server for the particular architecture. These files are discussed in more detail in Sections below.

Note: Additional FLEXlm documentation regarding the setup and use of the FLEXlm license server is provided in the vrand/flexlm/htmlman directory.

5.2 Starting the FLEXlm License Server on UNIX

After installing the FLEXlm license server on any of the supported UNIX machines, the *vrand/flexlm/$ARCH* directory will contain the following files:

Filename	Description
VRAND	VR&D FLEXlm vendor daemon
lmgrd	FLEXlm license server
lmutil	FLEXlm utilities

The *lmgrd* FLEXlm license server uses the VRAND vendor daemon to serve licenses for all VR&D products. The *lmutil* program provides the user with a set of utility programs for controlling the license server.

The user can use these programs directly, but VR&D provides a set of wrapper functions to aid the use of FLEXlm for VR&D products. These wrapper functions are provided in the form of shell scripts in the *vrand/flexlm* directory and will work on any of the supported UNIX platforms. These wrapper functions are summarized below:

Filename	Description
lmgrd	VR&D wrapper for the FLEXlm license server
lmutil	VR&D wrapper for the FLEXlm utilities functions

To start the license server the user would typically execute the *vrand/flexlm/lmgrd* VR&D wrapper for the FLEXlm license server as follows

```
lmgrd [-c <license file>] [-l <log file>]
```

where [-c <license file>] and [-l <log file>] are optional arguments that point to the appropriate VR&D license file and corresponding server log file. If no [-c <license file>] option is provided, *lmgrd* will try to use the default VR&D license file in the *vrand/licenses* directory. If no [-l <log file>] option is provided, *lmgrd* will use *vrand/flexlm/lmgrd_ $HOSTNAME.log* as the default log file.

To control the license server, the *lmutil* wrapper is provided. The *lmutil* options are obtained by simply typing

```
lmutil
```

at the command line. The options are summarized in the listing below:

```
lmutil - Copyright (c) 1989-2003 by Macrovision Corporation. All rights reserved.

usage:  lmutil lmborrow -status

        lmutil lmborrow -clear

        lmutil lmborrow {all|vendor} dd-mmm-yyyy:[time]
        lmutil lmborrow -return [-c licfile] [-d display_name] feature
        lmutil lmdiag [-c licfile] [-n]
        lmutil lmdown [-c licfile] [-q] [-all] [-vendor name] [-force] [-help]
        lmutil lmhostid [-internet|-user|-display|-n|
        -hostname|-string|-long]
        lmutil lminstall [-i infile] [-o outfile]
          [-overfmt {2, 3, 4, 5, 5.1, 6, 7.1, 8}]
          [-odecimal] [-maxlen n]
        lmutil lmnewlog [-c licfile] vendor new-file, or
        lmutil lmnewlog [-c licfile] feature new-file
        lmutil lmpath -status
        lmutil lmpath -override {all | vendor} path
        lmutil lmpath -add {all | vendor} path
        lmutil lmremove [-c licfile] feature user host display
        lmutil lmremove [-c licfile] -h feature host port handle
        lmutil lmreread [-c licfile] [-vendor name] [-all]
        lmutil lmswitchr [-c licfile] vendor new-file, or
        lmutil lmswitchr [-c licfile] feature new-file
        lmutil lmstat [-c licfile] [lmstat-args]
        lmutil lmswitch [-c licfile] vendor new-file, or
        lmutil lmswitch [-c licfile] feature new-file
        lmutil lmver flexlm_binary
        lmutil -help (prints this message)
        lmutil utility_name -help (display detailed usage information)
```

The VR&D wrapper for *lmutil* will automatically set the license file to the default VR&D license file in the *vrand/licenses* directory if no *-c licfile* option is provided. To use a specific utility, either specify the utility name and options to *lmutil*, for example

```
lmutil lmhostid
```

or copy the *lmutil* file to the name of the utility and execute the resulting file with the options for that utility, for example:

```
cp lmutil lmhostid
lmhostid
```

Links to a number of the most often used *lmutil* utilities are provided in the *vrand/bin* and *vrand/flexlm* directories.

5.3 Starting the FLEXlm License Server on Windows

After installing the FLEXlm license server on a Windows NT, 2000, or XP machine, the *vrand/flexlm/win32* directory will contain the following files:

Filename	Description
`VRAND.exe`	`VR&D FLEXlm vendor daemon`
`lmgrd.exe`	`FLEXlm license server`
`lmutil.exe`	`FLEXlm utilities`
`lmtools.exe`	`FLEXlm utilities with GUI`

Similar to the UNIX installation, the *lmgrd.exe* FLEXlm license server uses the *VRAND.exe* vendor daemon to serve licenses for VR&D products, while the *lmutil.exe* program provides the user with a set of utility programs for controlling the license server. See Section **5.2** for a detailed description of using *lmutil.exe*.

Unlike the UNIX installation, no wrapper programs are provided for the license server and utility programs on Windows platforms. Instead, VR&D provides the *lmtools.exe* program, which provides an easy to use graphical user interface to all the FLEXlm utilities. For Windows platforms, it is recommend to use the *lmtools.exe* program rather than the *lmutil.exe* program.

Note: To avoid problems with starting the FLEXlm license server, install it in a directory without space characters in the path name.

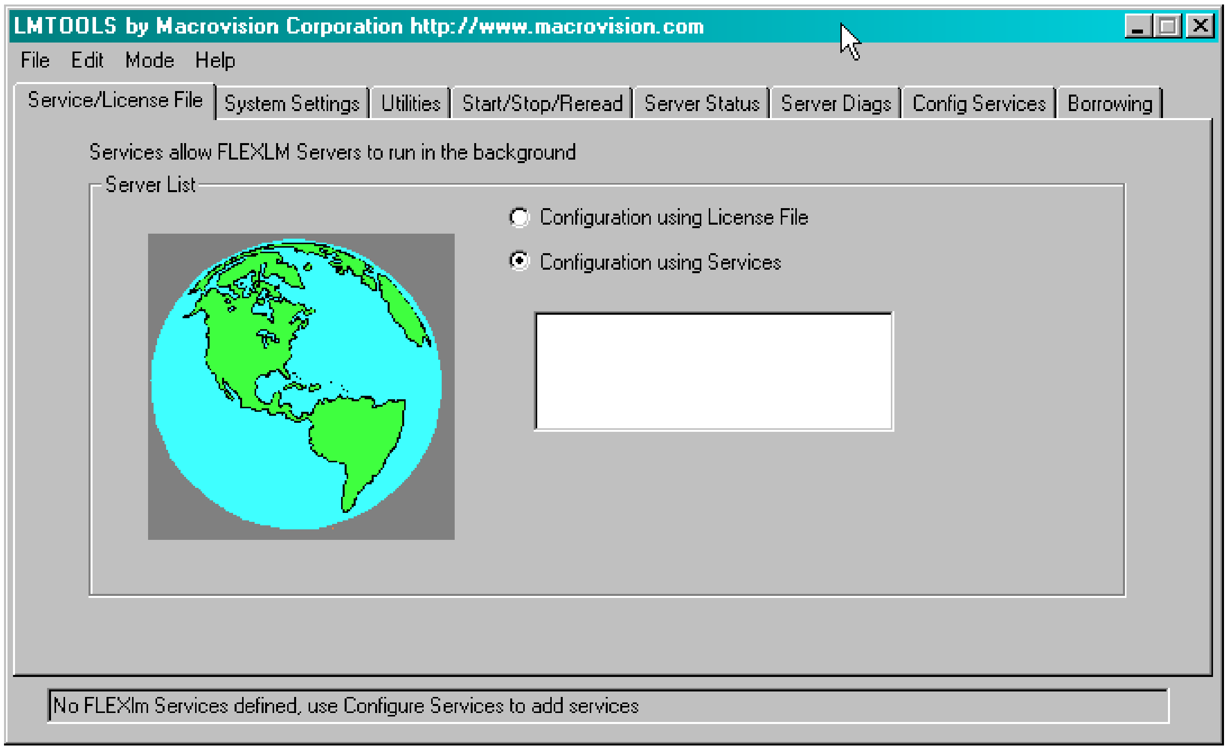
To start the license server on Windows platforms, make use of the *lmtools.exe* program. This will require several steps as outlined below:

Start the *lmtools.exe* program

To start *lmtools.exe*, simply double click on the *lmtools.exe* file in the *vrand/flexlm/win32* directory. This should open the *lmtools.exe* program as shown in **Figure 5-2**.

Install the license server as a service

Before starting the license server, it must be installed as a Windows NT service. The first step is to select the **Configuration using Services** option in **Figure 5-2**.



Services allow FLEXLM Servers to run in the background

Server List

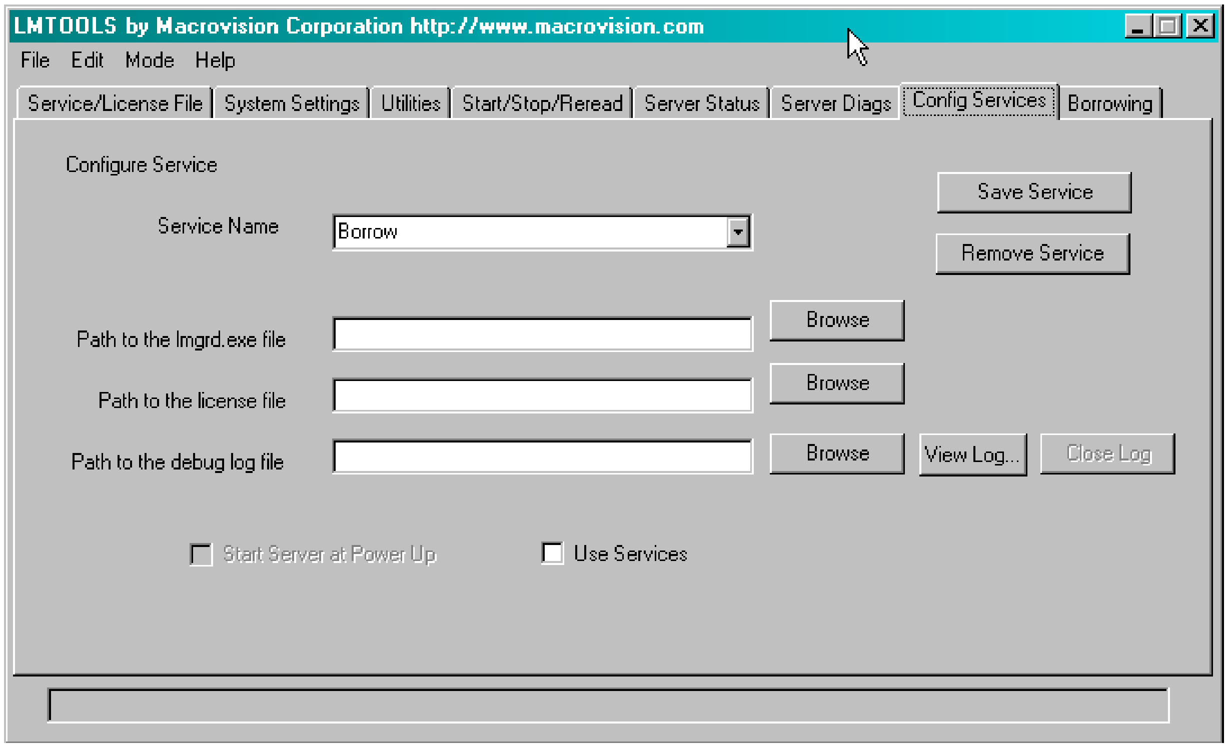
( ) Configuration using License File

(●) Configuration using Services

No FLEXlm Services defined, use Configure Services to add services

Figure 5-2LMTOOLS startup screen

Next select the **Configure Services** tab to obtain the window shown in **Figure 5-3**. The **Configure Services** tab will only appear after selecting the **Configuration using Services** option.



Configure Service

Service Name: Borrow

Path to the lmgrd.exe file

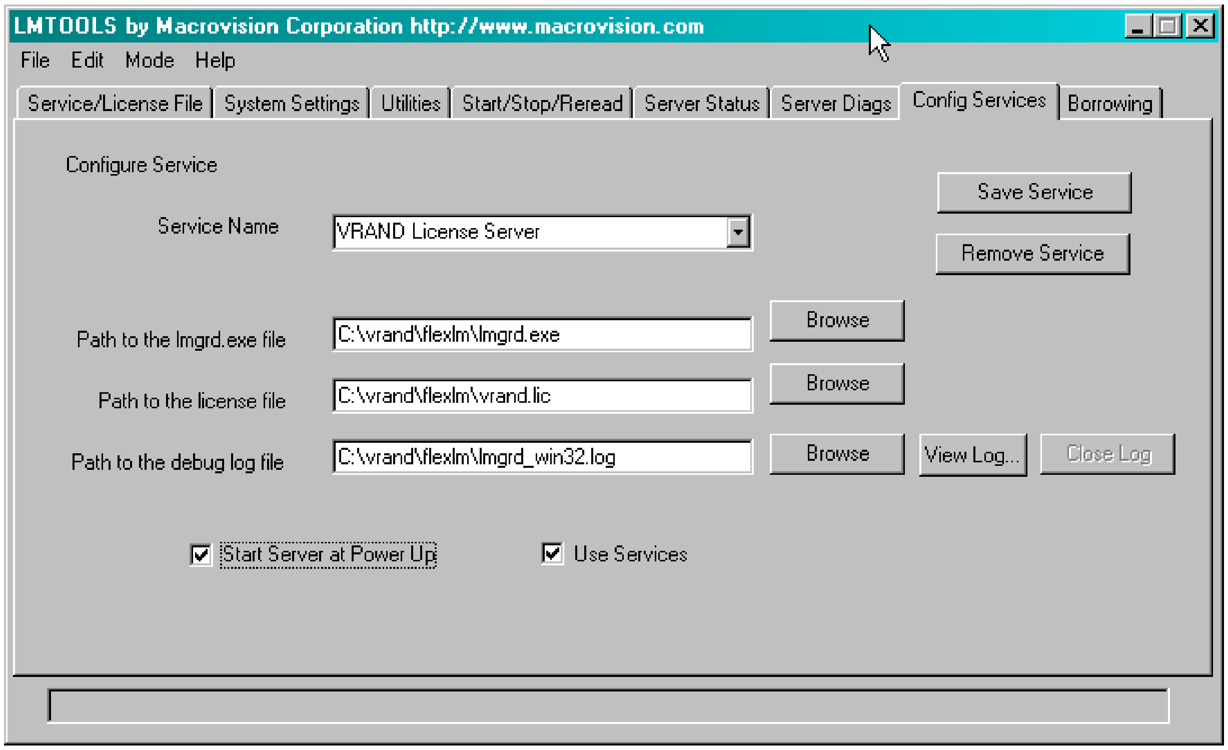
Path to the license file

Path to the debug log file

☐ Start Server at Power Up

☐ Use Services

Figure 5-3LMTOOLS service configuration screen


LMTOOLS by Macrovision Corporation http://www.macrovision.com
File Edit Mode Help
Service/License File | System Settings | Utilities | Start/Stop/Reread | Server Status | Server Diags | Config Services | Borrowing
Save Service
Remove Service
Browse
Browse
Browse
View Log...
Close Log

Configure Service

Service Name: VRAND License Server

Path to the lmgrd.exe file: C:\vrand\flexlm\lmgrd.exe

Path to the license file: C:\vrand\flexlm\vrand.lic

Path to the debug log file: C:\vrand\flexlm\lmgrd_win32.log

☑ Start Server at Power Up ☑ Use Services

Figure 5-4Updated LMTOOLS service configuration screen

Enter the license service name, **VRAND License Server**, in the **Service Name** text field and browse for the appropriate **lmgrd.exe**, the VR&D license file and the log file paths. For a typical installation, the **Configure Service** windows should appear as shown in **Figure 5-4**.

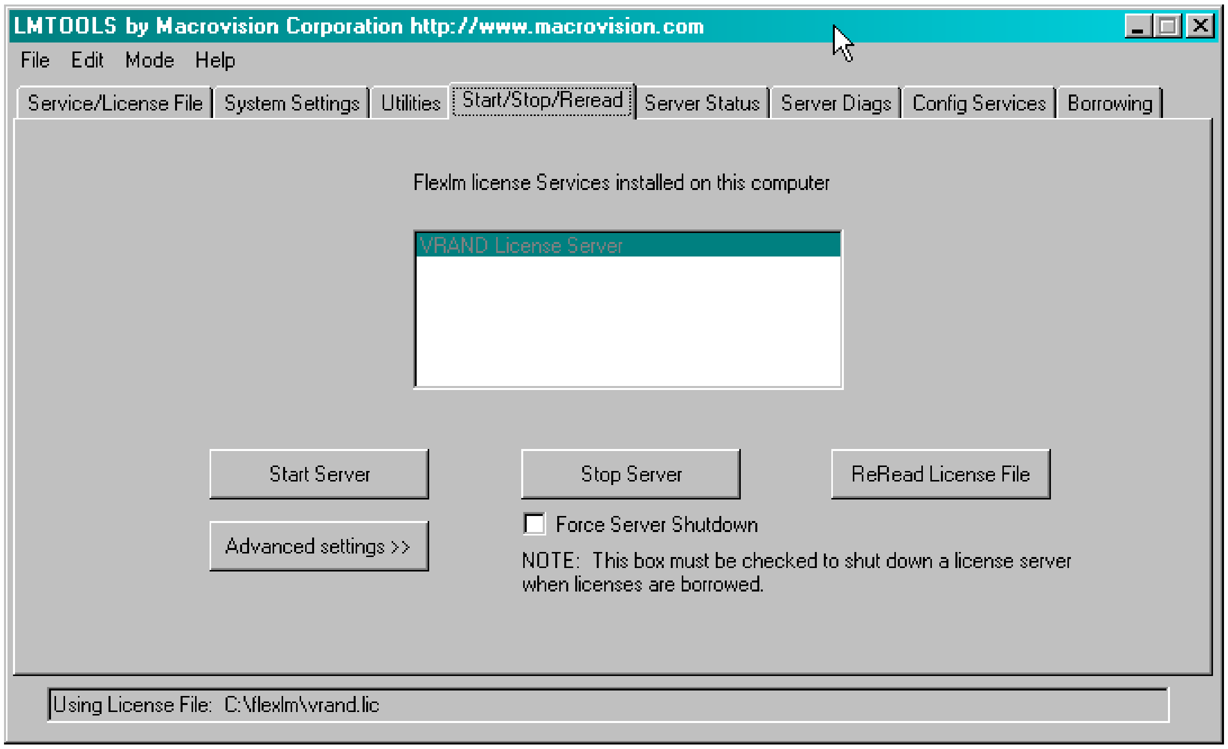
Make sure to select the **Start Server at Power Up** and **Use Services** options as shown in **Figure 5-4**. These two options will make sure the server is installed as a Windows NT service that will automatically start each time the machine is restarted. Finally, select the **Save Service** button to install the service.

Start the license server

The license server is now installed as a Windows NT service and all that remains is to start the server. Select the **Start/Stop/Reread** tab to obtain the window shown in **Figure 5-5**.

To start the license server, select the **Start Server** button.

The license server should now be started and the *lmtools.exe* software can be used to maintain the server. For example, *lmtools.exe* can check the status of the server, start and stop the server, force the server to re-read a new license file, etc.



Flexlm license Services installed on this computer

VRAND License Server

NOTE: This box must be checked to shut down a license server when licenses are borrowed.

Using License File: C:\flexlm\vrand.lic

Figure 5-5LMTOOLS server control screen

6 Contact Info

Please feel free to contact VR&D with any questions and/or comments, using any of the following contact information:

Vanderplaats Research and Development, Inc.

1767 S. 8th Street	**41700 Gardenbrook Rd. Suite 115**
Colorado Springs, CO 80905	**Novi, MI 48375**
Tel: +1-719-473-4611	**Tel: +1-248-596-1611**
Fax: +1-719-473-4638	**Fax: +1-248-596-1911**

Web Site: www.vrand.com
Email: visualdoc.support@vrand.com

7 Appendix A: Mounting the CD-ROM Drive on Supported UNIX Systems

On all UNIX platforms, it may be necessary to mount the CD-ROM drive before installing the VisualDOC software from the CD-ROM. Mounting the CD-ROM drive on any of the supported UNIX platforms is discussed below. After the CD-ROM drive is mounted, the installation is started by issuing the following command

```
/cdrom/install.sh
```

Note: In the discussion below, # represents the SCSI ID of the CD-ROM device and the VR&D CD-ROM needs to be in the drive before mounting it.

Solaris

For Solaris 2.7 or higher, the system should automatically mount the CD-ROM in */cdrom/cd-rom0*. If the system did not mount the CD-ROM automatically, mount the CD-ROM using the following commands:

```
mkdir /cdrom
mount -F hsfs -r /dev/dsk/c0t#d0s0 /cdrom
```

Linux 32 and 64 Bit

The system should automatically mount the CD-ROM. If the system did not mount the CD-ROM automatically, mount the CD-ROM using the following commands:

```
mkdir /cdrom
mount -t iso9660 /dev/cdrom /cdrom
```

8 Appendix B: Setting/Editing Environment Variable on Windows Computers

The procedure for setting/editing environment variables is slightly different for the various Windows configurations. Each Windows configuration is considered separately for the case of adding the *<install_dir>/vrand/vdoc6.2/WIN32* directory to the PATH environment variable.

Windows XP

On Windows XP computers, environment variables are added/modified by selecting the **Performance & Maintenance** option from the **Control Panel**. Next select the **System** options which will open the **System Properties** dialog box, where the **Advanced** tab should be selected. Finally, select the **Environment Variables** button on the **Advanced** tab. To edit the PATH environment variable, highlight the variable and select the **Edit** button. If the PATH environment variable is edited from the **User Variables** list, it will be modified only for the current user. If the PATH environment variable is edited from the **System Variables** list, it will be modified for all users.

After selecting the appropriate **Edit** button, a dialog box will appear for editing the PATH environment variable. Simply add the *<install_dir>/vrand/vdoc6.2/WIN32* directory to the existing value of the *PATH* environment variable. Note that multiple directories are separated by a ";"character. To add a new environment variable, select the appropriate **New** instead of the **Edit** button.

Windows 2000

On Windows 2000 computers, environment variables are added/modified by selecting the **System** option from the **Control Panel**. This will open the **System Properties** dialog box, where the **Advanced** tab should be selected. Finally, select the **Environment Variables** button on the **Advanced** tab. To edit the PATH environment variable, highlight the variable and select the **Edit** button. If the PATH environment variable is edited from the **User Variables** list, it will be modified only for the current user. If the PATH environment variable is edited from the **System Variables** list, it will be modified for all users.

After selecting the appropriate **Edit** button, a dialog box will appear for editing the PATH environment variable. Simply add the *<install_dir>/vrand/vdoc6.2/WIN32* directory to the existing value of the *PATH* environment variable. Note that multiple directories are separated by a ";"character. To add a new environment variable, select the appropriate **New** instead of the **Edit** button.