CROSSGRID INSTALLATION GUIDE

TASK 2.2 MARMOT

WP 2.2- Code Debugging and Verification (MARMOT)

Document Filename:	**CG2.2-v0.3-UST002-MARMOTInstallGuide-v1.1.11.doc**
Work package:	**WP 2.2- Code Debugging and Verification (MARMOT)**
Partner(s):	**USTUTT, CSIC**
Lead Partner:	**USTUTT**
Config ID:	**CG2.2-v0.3-UST002-MARMOTInstallGuide-v1.1.11**
Document classification:	**PUBLIC**

Abstract:
This is the installation guide for the MPI analysis and checking tool MARMOT.

Document Log

Version	Date	Summary of changes	Author
0-1	13/08/2004	Skeleton und first chapters on integration plan.	Bettina Krammer
0.2	19/10/2004	Adapted template for marmot. First draft.	Bettina Krammer
0.3	19/10/2004	Included reviewer's comments (reviewer: Christos Kanellopoulos)	Bettina Krammer
	26/01/2005	Verified by the QE	Robert Pajak

CONTENTS

COPYRIGHT NOTICE

This research is partly funded by the European Commission IST-2001-32243 Project “CrossGrid”.

1. ABOUT THE SOFTWARE

- MARMOT is a library written in c++, which has to be linked to your application in addition to the existing MPI library.
- It will check if your application conforms to the MPI standard and will issue warnings if there are errors or non-portable constructs.
- You need not modify your source code, you only need one additional process working as MARMOT's debug server.
- MARMOT's output is a human-readable log file.
- The tool can be configured via environment variables.
- Currently MPI-1.2 standard is supported.
- See also the README in MARMOT's distribution.
- MARMOT's source code can be found at

http://savannah.fzk.de/cgi-bin/viewcvs.cgi/crossgrid/crossgrid/wp2/wp2_2-verif/src/MARMOT/

- MARMOT's RPMs can be found at

http://gridportal.fzk.de/distribution/crossgrid/autobuilt/i386-rh7.3-gcc3.2.2/wp2/RPMS/

The latest tag is currently v1.1.11.

1.1. SOFTWARE COMPONENTS

- For CrossGrid, there are two distributions, namely cg-wp2.2-marmot for mpich-p4 and cg-wp2.2-marmot-g2 for mpich-g2.
- The following files will be installed in the following default directories:
  - LIB/libdmpi.a, LIB/libmpo.a, LIB/libfmpo.a will be installed in @prefix@/lib/MARMOT
    These are the MARMOT libraries that have to be linked to an application to make use of the tool.
  - README, CONFIG-EXAMPLES, DEPENDENCIES, INSTALL, USERS_GUIDE will be installed in @prefix@/share/doc/MARMOT
    This is the most important documentation for MARMOT. (USERS_GUIDE is not contained in tag v1_1_11)
  - TEST_C/basic, TEST_C/basic.jdl, TEST_C/basic-g2.jdl, TEST_C/cg-tutorial-marmot-exercise, TEST_C/cg-tutorial-marmot-exercise.jdl will be installed in @prefix@/bin/MARMOT/C
    These are simple test programs written in C to test MARMOT's functionality.
  - TEST_F/basic, TEST_F/basic.jdl, TEST_F/basic-g2.jdl will be in prefix@/bin/MARMOT/F.
    These are simple test programs written in Fortran to test MARMOT's functionality.
  - The default for @prefix@ is /usr/local.
  - The rpms for CrossGrid will install files to the /opt/cg directory, i.e. @prefix@ = /opt/cg. The subdirectories will be named cg-wp2.2-marmot and cg-wp2.2-marmot-g2 instead of MARMOT.
- The MARMOT packages should be deployed on developer's workstations, UI (for relinking of applications), CE and WNs (for running tests and dynamically linked applications).
- See also the INSTALL file in MARMOT's distribution.

1.2. DEPENDENCIES

- See also the DEPENDENCIES file in MARMOT's distribution.
- MPI implementation
  - Since the application that is to be verified is written using MPI, the MPI library is needed to run the application. MARMOT verifies the calls made by the program with the use of the so called profiling interface. This profiling interface is part of the MPI standard. Any MPI implementation that conforms to the MPI standard needs to provide this interface. Therefore, this requirement should not limit the selection of possible MPI libraries.
  - The MPI implementation itself may require some other software, for example globus libraries when using mpich-g2.
  - Some of MARMOT's source files include mpi.h, therefore mpi (i.e. at least mpi.h) is needed for the compilation of MARMOT to make the MARMOT libraries, which can then be linked to an application.
- C++ compiler:
  - MARMOT is implemented in C++. The compiler should implement the ISO/IEC 14882 language specification of C++. We have succeeded in using gcc 2.96 or later, earlier versions might not work properly. Intel Compilers are an alternative, they are available for no cost for non-commercial use on linux platforms. For example, on our local environment, Intel compilers version 7.0 or the KAI C++ 4.0 compiler have been used successfully. MARMOT is also tested with xlC_r for AIX Compiler, Version 6.
  - To link MARMOT to a C or Fortran application with the C/Fortran linker instead of the c++ linker, some c++ libraries will have to be linked, too (for example libstdc++, using gcc or g77).
- Fortran Compiler:
  - To support the Fortran binding of the MPI standard a Fortran compiler is required. The same Fortran compiler should be used to compile the application.
- C compiler:
  - To support C applications, a C compiler is required (any C compiler should do).
- Other tools that users may need for installation
  - make (tested with GNU make 3.79.1 and later versions) for compilation.
- Other tools that users do not necessarily need
  - Doxygen (tested with version 1.2.14 and later versions) is used to automatically generate documentation.
  - MARMOT's distribution comes with all the required files that users just have to run "configure" and then "make" in order to build the tool. Users do not need autotools.

2. INSTALLATION IN THE CROSSGRID TESTBED

The CrossGrid testbeds are managed by the LCFG deployment support tool. This tool allows an automatic installation of the software on all required nodes.

2.1. RPM LISTS FOR LCFG

In order to install MARMOT via LCFG the user needs to make some additions to the rpm lists for the developer workstation (DeveloperWorkstation-CG-rpm.h), the computing element (ComputingElement-GC-rpm.h) and the worker node (WorkerNode-CG-rpm.h). In all of the above rpm lists you should add the following lines:

cg-wp2.2-marmot-1.1.11-1

cg-wp2.2-marmot-g2-1.1.11-1

2.2. PROFILE MODIFICATIONS FOR LCFG

No LCFG profiles need to be modified.

2.3. MANUAL POST INSTALLATION STEPS

No extra post installation steps are required.

3. MANUAL INSTALLATION

3.1. DOWNLOAD

The software installation using RPMs just requires downloading the RPMS, for example

- wget http://gridportal.fzk.de/distribution/crossgrid/autobuilt/i386-rh7.3-gcc3.2.2/wp2/RPMS/cg-wp2.2-marmot-1.1.11-1.i386.rpm
- wget http://gridportal.fzk.de/distribution/crossgrid/autobuilt/i386-rh7.3-gcc3.2.2/wp2/RPMS/cg-wp2.2-marmot-g2-1.1.11-1.i386.rpm

The latest tag is currently v1.1.11.

3.2. INSTALLATION FROM RPM

MARMOT can then be installed by

- rpm –ivh cg-wp2.2-marmot-1.1.11-1.i386.rpm
- rpm –ivh cg-wp2.2-marmot-g2-1.1.11-1.i386.rpm

3.3. INSTALLATION FROM SOURCE

Installation by configure and make is also possible, for details see also the INSTALL file and the CONFIG-EXAMPLES file in MARMOT's distribution.

- A download version of MARMOT's source code can be found at

http://savannah.fzk.de/distribution/crossgrid/autobuilt/i386-rh7.3-gcc3.2.2/wp2/SOURCES/cg-wp2.2-marmot-1.1.11.tar.gz

- Download the source code, go to the directory MARMOT and run the autogen.sh script
  ```
  $ ./autogen-sh
  ```
  to create the Makefile.in from the Makefile.am,
  the configure script from the configure.ac etc.
- Run
  ```
  $ ./configure  --with-mpi-dir=/opt/cg/mpich  --enable-mpichp4  \
  CXX=/opt/gcc-3.2.2/bin/g++-3.2.2 \
  CC=/opt/gcc-3.2.2/bin/gcc-3.2.2 \
  F77=/opt/gcc-3.2.2/bin/g77-3.2.2
  ```
  or
  ```
  $ ./configure --with-mpi-dir=/opt/cg/mpich \
  --enable-globus --with-globus-dir=/opt/cg/globus \
  CXX=/opt/gcc-3.2.2/bin/g++-3.2.2 \
  CC=/opt/gcc-3.2.2/bin/gcc-3.2.2 \
  F77=/opt/gcc-3.2.2/bin/g77-3.2.2
  ```
  to create the Makefiles etc. automatically from the corresponding
  templates named *.in. The first configure options are for creating the mpich-p4 version, the second ones for creating the mpich-g2 version of MARMOT.
- Run
  ```
  make
  ```
- Run
  ```
  make install DESTDIR=/opt/cg
  ```
  The following files will be installed in the following default directories:
  - LIB/libdmpi.a, LIB/libmpo.a, LIB/libfmpo.a will be installed in /opt/cg/lib/MARMOT

These are the MARMOT libraries that have to be linked to an application to make use of the tool.

- README, CONFIG-EXAMPLES, DEPENDENCIES, INSTALL, USERS_GUIDE will be installed in /opt/cg/share/doc/MARMOT
  This is the most important documentation for MARMOT.
- TEST_C/basic, TEST_C/basic.jdl, TEST_C/basic-g2.jdl, TEST_C/cg-tutorial-marmot-exercise, TEST_C/cg-tutorial-marmot-exercise.jdl will be installed in /opt/cg/bin/MARMOT/C
  These are simple test programs written in C to test MARMOT's functionality.
- TEST_F/basic, TEST_F/basic.jdl, TEST_F/basic-g2.jdl will be in /opt/cg/bin/MARMOT/F.
  These are simple test programs written in Fortran to test MARMOT's functionality.
- **Note:** Unlike installation from RPM, subdirectories are named MARMOT and not cg-wp2.2-marmot or cg-wp2.2-marmot-g2 as it should be.

3.4. CONFIGURATION

3.4.1. List of configuration files

MARMOT does not use any configuration files, so no editing of configuration files is needed.

3.4.2. Editing the configuration files

3.4.3. Startup scripts

MARMOT does not use startup scripts.

3.4.4. Other requirements

3.4.4.1. Environment

- The default behaviour of MARMOT can be overridden by setting special environment variables.
- NOTE: In previous releases (<= v1.1.11), these variables were not prefixed with "MARMOT_", i.e. later releases use MARMOT_DEBUG_MODE instead of DEBUG_MODE.
- In general, it depends on your system how to set environment variables, e.g. if you simply have to use a commandline like in bash
  export MARMOT_DEBUG_MODE=1
  or in other shells
  setenv MARMOT_DEBUG_MODE 1
  or if you have to edit a script like .bashrc or others.
- See also the USERS_GUIDE file in MARMOT's distribution.
- MARMOT_DEBUG_MODE
  Set the environmental variable MARMOT_DEBUG_MODE (integer) for the debug mode:
  - MARMOT_DEBUG_MODE < 0: nothing is reported.
  - MARMOT_DEBUG_MODE = 0: only errors
  - MARMOT_DEBUG_MODE = 1: errors and warnings
  - MARMOT_DEBUG_MODE = 2: errors, warnings and remarks are reported.

  The default value is 2.
- MARMOT_INTERFACE_MODE

NOTE: THE FOLLOWING HOLDS TRUE FOR OLD VERSIONS OF MARMOT: MEANWHILE, MARMOT_INTERFACE_MODE IS SET AUTOMATICALLY TO THE CORRECT VALUE.
Set the environmental variable MARMOT_INTERFACE_MODE for the interface mode (default 0). The values mean

- 0: C Interface
- 1: Fortran interface

It does not do any harm to run a Fortran program with C INTERFACE_MODE=0, in this case, MARMOT just may issue incorrect warnings like "ERROR: datatype is Fortran-Type" if one uses e.g. MPI_INTEGER (which is correct in a Fortran program but not in a C program).

- MARMOT_SERIALIZE
  Set the environmental variable MARMOT_SERIALIZE for serializing the code or not.
  - MARMOT_SERIALIZE = 0: code is not serialized
  - MARMOT_SERIALIZE = 1: code is serialized

  default 0
- MARMOT_MAX_TIMEOUT_DEADLOCK
  Set the environmental variable MARMOT_MAX_TIMEOUT_DEADLOCK for the maximum time all calls are allowed to stay pending before a deadlock warning is issued, i.e. set a value in microseconds for MARMOT_MAX_TIMEOUT_DEADLOCK (default 1000000 microseconds = 1s).
- MARMOT_MAX_TIMEOUT_SERIALIZE
  Set the environmental variable MARMOT_MAX_TIMEOUT_SERIALIZE for the maximum message time in case of serialization (MARMOT_SERIALIZE=1), i.e. set a value in microseconds for MARMOT_MAX_TIMEOUT_SERIALIZE (default 1000 microseconds).
- MARMOT_TRACE_CALLS
  Set the environmental variable MARMOT_TRACE_CALLS, whether calls shall be traced back or not (default 1). The values mean
  - 1: calls are traced with output to stderr, traceback in case of a deadlock is possible
  - 0: calls are traced without output to stderr, traceback in case of a deadlock is possible
  - -1: calls are not traced, traceback in case of a deadlock is NOT possible.

  The number of calls to be traced back in case of deadlock can be set via MARMOT_MAX_PEND_COUNT.
- MARMOT_MAX_PEND_COUNT
  Set the environmental variable MARMOT_MAX_PEND_COUNT for the maximum number of MPI calls that can be traced back, i.e. set an int value for MARMOT_MAX_PEND_COUNT (default 10).

3.4.4.2. Users

There is no special user needed.

3.4.4.3. Ports

There are no open ports needed (unless the application to be checked needs them).

3.4.4.4. Certificates

MARMOT does not require any certificates.

3.4.4.5. Folders

MARMOT does not require any special folders.

4. RUNNING AND TESTING

- MARMOT's source code contains two subdirectories TEST_C and TEST_F with simple MPI test programs written in C/Fortran. Some of these programs are correct, some of these programs are erroneous to evoke MARMOT's warnings. Users may play with these programs to get a taste of MARMOT.
- For the Crossgrid tutorial there is a special program TEST_C/cg-tutorial-marmot-exercise.c.
- Any application from WP 1 might be used but will not evoke enough warnings from MARMOT to really demonstrate its functionality.
- After installation, the directories /opt/cg/bin/cg-wp2.2-marmot/C and /opt/cg/bin/cg-wp2.2-marmot/F and /opt/cg/bin/cg-wp2.2-marmot-g2/C and /opt/cg/bin/cg-wp2.2-marmot-g2/F contain some example binaries, see section 1.1 about the files to be installed.
- To run the application with MARMOT, one has to add an additional process working as debug server, i.e. one needs (n+1) instead of n processes

  $ mpirun -np (n+1) foo

  MARMOT's output is sent to stderr.
- For the use on the testbed, the simple test program basic is installed in /opt/cg/bin/cg-wp2.2-marmot/C. It only performs MPI_Init and MPI_Finalize. Use for example a jdl file like the following basic.jdl:

```
Executable          = "basic";
JobType             = "MPICH";
NodeNumber          = 3;
VirtualOrganisation = "cg";
StdOutput           = "basic.out";
StdError            = "$HOME/basic.err";
InputSandbox        = {"basic"};
OutputSandbox       = {"basic.out","$HOME/basic.err"};
```

  to submit your job via Resource Broker

  $ edg-job-submit basic.jdl

  You can watch the output with something like

  $ tail –f basic.err

  on the CE where the job runs, and later on, you can get the output with edg-job-get-output. Same for mpich-g2 jobs.
- Applications written in C can be compiled accordingly to this example:

  gcc -I/opt/cg/mpich/include -c basic.c
  g++ -o basic basic.o -L../LIB -ldmpi -lmpo -L/opt/cg/mpich/lib -lmpich

  (with gcc = /opt/gcc-3.2.2/bin/gcc-3.2.2, g++ = /opt/gcc-3.2.2/bin/g++-3.2.2)

  It is also possible to use mpicc, in this case it is very important to link the proper version of the libstdc++ (i.e. same version as was used to compile the MARMOT libraries, i.e. specify the correct path for -lstdc++):

  mpicc -c basic.c
  mpicc -o basic basic.o -L../LIB -ldmpi -lmpo -L/opt/cg/mpich/lib -lmpich \
  -L/opt/gcc-3.2.2/lib -lstdc++
- Applications written in Fortran can be compiled accordingly to this example:
  (with g77 = /opt/gcc-3.2.2/bin/g77-3.2.2, g++ = /opt/gcc-3.2.2/bin/g++-3.2.2):

  g77 -I/opt/cg/mpich/include -g -O2 -c basic.f
  g++ -I/opt/cg/mpich/include -g -O2 -o basic basic.o \

```
-L../LIB -ldmpi -lfmpo -lmpo \
-L/opt/cg/mpich/lib -lmpich  \
-L/opt/gcc-3.2.2/lib/gcc-lib/i686-pc-linux-gnu/3.2.2 \
-L/opt/gcc-3.2.2/lib/gcc-lib/i686-pc-linux-gnu/3.2.2/../../.. -lfrtbegin \
-lg2c -lm -lgcc_s
```

It is also possible to use mpif77, in this case it is very important to link the proper version of the libstdc++ (i.e. same version as was used to compile the MARMOT libraries, i.e. specify the correct path for -lstdc++):

```
mpif77 -c basic.f
mpif77 -o basic basic.o -L../LIB -ldmpi -lfmpo -lmpo  \
-L/opt/cg/mpich/lib -lmpich \
-L/opt/gcc-3.2.2/lib -lstdc++
```

- See also the USERS_GUIDE and CONFIG-EXAMPLES file in MARMOT's distribution.

4.1. LOG FILES

- MARMOT produces a human-readable log file.
- By default, MARMOT's logging is directed to stderr, users may redirect this output wherever they like.
- The log file will start with a header to give you an overview of the environment variables you set:

```
=============== environmental variables ====================
MARMOT_DEBUG_MODE=2
    (0: errors,
     1: errors and warnings,
     2: errors, warnings and remarks are reported,
     default: 2)
MARMOT_INTERFACE_MODE=0
    (0: C interface,
     1: Fortran interface,
     interface mode is set automatically)
MARMOT_SERIALIZE=0
    (0: code is not serialized,
     1: code is serialized,
     default: 0)
MARMOT_TRACE_CALLS=1
    (1: calls are traced with output to stderr,
        traceback in case of a deadlock is possible,
     0: calls are traced without output to stderr,
        traceback in case of a deadlock is possible,
    -1: calls are not traced,
        traceback in case of a deadlock is NOT possible.
     default: 1
     Number of calls to be traced back in case of deadlock
     can be set via MAX_PEND_COUNT.)
MARMOT_MAX_PEND_COUNT=10
    (maximum number of calls that are traced back,
     default: 10)
MARMOT_MAX_TIMEOUT_DEADLOCK=1000000
    (maximum message time,
```

```
     default: 1000000 microseconds)
MARMOT_MAX_TIMEOUT_SERIALIZE=1000
    (maximum message time,
     default: 1000 microseconds)
MARMOT_MAX_TIMEOUT_ONE=0
    (maximum message time,
     default: 0 microseconds)
MARMOT_MAX_TIMEOUT_TWO=0
    (maximum message time,
     default: 0 microseconds)
============================================================
```

- For example, the output of the cg-tutorial-marmot-exercise will contain warnings such as

```
18 rank 2 performs MPI_Type_struct
WARNING: MPI_Type_struct: blocklength[0]=0!
WARNING: MPI_Type_struct: datatype[0] is Fortran-Type!
WARNING: MPI_Type_struct: datatype[1] is optional!
WARNING: MPI_Type_struct: blocklength[0]=0!
WARNING: MPI_Type_struct: blocklength[0]=0!
19 rank 0 performs MPI_Type_struct
20 rank 1 performs MPI_Type_struct
21 rank 2 performs MPI_Type_struct
WARNING: MPI_Type_struct: datatype[0] is Fortran-Type!
WARNING: MPI_Type_struct: datatype[1] is optional!
WARNING: MPI_Type_struct: datatype[0] is Fortran-Type!
WARNING: MPI_Type_struct: datatype[1] is optional!
22 rank 0 performs MPI_Type_commit
23 rank 1 performs MPI_Type_commit
24 rank 0 performs MPI_Type_commit
25 rank 1 performs MPI_Type_commit
26 rank 2 performs MPI_Type_commit
NOTE: MPI_Type_commit: Datatype already committed!
NOTE: MPI_Type_commit: Datatype already committed!
27 rank 0 performs MPI_Address
28 rank 1 performs MPI_Address
29 rank 2 performs MPI_Type_commit
NOTE: MPI_Type_commit: Datatype already committed!
30 rank 0 performs MPI_Address
31 rank 1 performs MPI_Address
32 rank 2 performs MPI_Address
33 rank 0 performs MPI_Type_struct
34 rank 1 performs MPI_Type_struct
35 rank 2 performs MPI_Address
36 rank 0 performs MPI_Type_commit
37 rank 1 performs MPI_Type_commit
38 rank 2 performs MPI_Type_struct
39 rank 0 performs MPI_Issend
40 rank 1 performs MPI_Issend
41 rank 2 performs MPI_Type_commit
WARNING: MPI_Issend: count=0 !
WARNING: MPI_Issend: datatype is for reduction functions!
WARNING: MPI_Issend: count=0 !
WARNING: MPI_Issend: datatype is for reduction functions!
42 rank 0 performs MPI_Recv
```

```
WARNING: MPI_Recv: count = 0!
43 rank 1 performs MPI_Recv
44 rank 2 performs MPI_Issend
```

The warning for `MPI_Issend` refers for example to

```
/* This will produce warnings:
 * the count is 0,
 * the types are for reduction functions.
 */
MPI_Issend(&int_send_buf, 0, MPI_LONG_INT, right, MSG_TAG,
           MPI_COMM_WORLD, &request);
```

where we had deliberately put some errors.

- More details can be found in MARMOT's tutorial description, see http://www.eu-crossgrid.org/wp5-1-login/CGTutorial.htm (password-protected).

5. THE GPL LICENSE AGREEMENT

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program).Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, does not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for non-commercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

6. BIBLIOGRAPHY