Installation of OpenPALM on Mac OS X

By The OpenPALM Team†

1. Download of the latest stable OpenPALM distribution

The latest stable release of OpenPALM is downloadable from the OpenPALM web site (Fig. 1): http://www.cerfacs.fr/globc/PALM_WEB/
The instructions to download OpenPALM are accessible from the page *Become a user*. In order to inform the OpenPALM Team of your download, you have to fill a form (Fig. 2).

2. Introduction

In the OpenPALM distribution you will find the source codes of the OpenPALM library, of its interface and of all the sessions of the training. The first thing to do is to decompress the gzipped tar archive of the distribution:

```
> tar -xvfz distrib.tgz
```

Two directories are created: PrePALM_MP and PALM_MP. The first one contains the graphical user interface PrePALM, the second one the OpenPALM library. Note that the OpenPALM manual is in *PrePALM/DOC* (in english as well as in french) and that the sources for the training sessions are in *PrePALM/training*

3. Installation of the PrePALM graphical user interface

3.1. *Pre-requirements*

The graphical interface PrePALM is written in Tcl/Tk with some C. Therefore you need these two environments on the machines where PrePALM has to run. The Tcl/Tk version has to be at least 8.3.

A small C program is used to interpret the STEPLANG language: it is therefore necessary to compile this component. A pre-compiled version working on i386 to i686 and x86_64 platforms is provided with the OpenPALM distribution. The most widespread public domain algebra libraries (such as BLAS, LAPACK, ScaLAPACK) interfaced in the OpenPALM algebra toolbox are not provided with the OpenPALM distribution and should be installed (if they are not already pre-installed) on the machines where the final application has to be compiled and executed. Additionally, the geophysical interpolation library based on the OASIS coupler and on the SCRIP algorithms is provided with the OpenPALM distribution.

3.2. *PrePALM command definition*

The graphical user interface is written in Tcl/Tk which is an interpreted language. Therefore there is no need of compilation. Nevertheless every user has to set an environment variable containing the installation path and an alias as a shortcut for the GUI. Accordingly to the preferred shell you should add to the .cshrc or .bashrc or .rc file:

csh, tcsh:

```
setenv PREPALMMPDIR path_to_PrePALM
alias prepalm $PREPALMMPDIR/prepalm_MP.tcl \!* &
```

sh, bash:

† CERFACS, 42 avenue G. Coriolis, 31 057 Toulouse Cedex 01, France

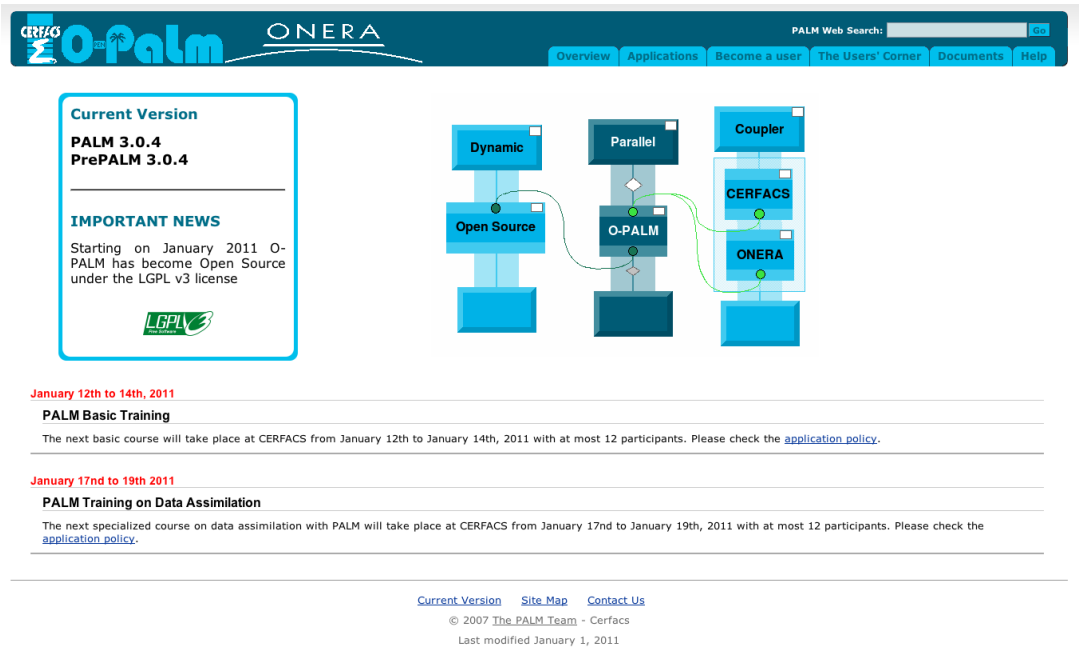

FIGURE 1. OpenPALM web site.

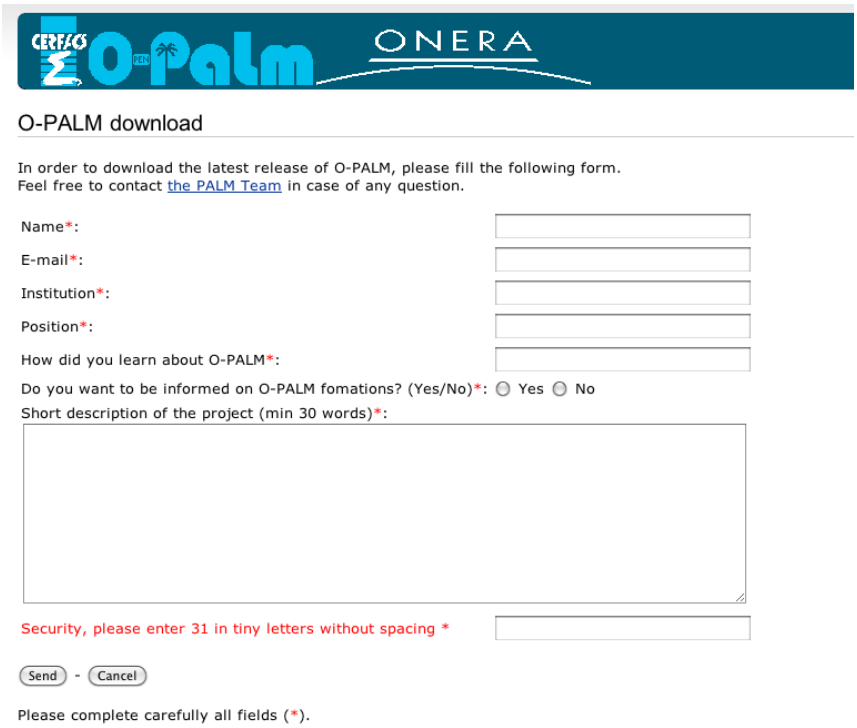

FIGURE 2. OpenPALM download page.

```
1 function prepalm {
2 export PREPALMMPDIR=path_to_PrePALM
3 $PREPALMMPDIR/prepalm_MP.tcl $* &
4 }
```

Optionally you can set the PREPALMEDITOR environment variable pointing to your preferred

editor that PrePALM will start every time it proposes to access an external editor. The default is vi. If you rather prefer emacs you should add to the shell configuration file:

csh, tcsh:

```
setenv PREPALMEDITOR emacs
```

sh, bash:

```
export PREPALMEDITOR=emacs
```

3.3. *STEPLANG interpreter installation*

Steplang is the command language used to describe the event driven actions manipulating the objects stored in the buffer. If you need to recompile its interpreter, enter STEPLANG the directory

```
> cd PrePALM_MP/STEPLANG/
```

Modify, if needed, the simple Makefile

```
CC=gcc
ARCH=osx
```

and compile:

```
> make clean
> make
```

If everything go right, you should end up with the steplang-osx executable creation.

4. Installation of the PALM library

4.1. *Pre-requirements*

The OpenPALM library includes the objects used to generate the OpenPALM driver (palm_main) and the user defined entities (units and blocks). This library has to be compiled on the platform where the OpenPALM applications will eventually run. The installation procedure is based on the automatic configuration tool autoconf. Remember that OpenPALM has been implemented in FORTRAN 90 and C. To install PALM it is therefore necessary to have access to:

- A FORTRAN 90 and a C compiler. They have to be compatible. The best idea is to use the two compilers from a same distribution and at the same version
- An MPI library that implements the MPI-2 standard (one does not need MPI-2 if he is only going to work in MPI-1 mode. Cf. Chapter 15 of the OpenPALM manual). The MPI library has to compiled with the same compiler as in the previous item.

Optionally, depending on the OpenPALM features you are going to use, you may need

- the standard scientific libraries BLAS and LAPACK (possibly optimised by the manufacturer).
- the parallel algebra libraries PBLAS and SCALAPACK
- the NetCDF I/O library
- the sources of the minimisers of which the interface is available in the OpenPALM algebra toolbox

Remark: you do not need superuser rights to install OpenPALM on your machine.

4.2. *Generalities concerning installation*

You install OpenPALM with simply three commands from the *PALM_MP* directory of the distribution:

```
> ./configure [OPTION]... [VAR=VALUE]...
> make clean
> make
> make install
```

The only step requiring some attention is the first one, for which you have to choose the proper options for the configuration. They depend on your compilers, on the platform, on the MPI distribution and, finally, on the flavor of OpenPALM (single proc, MPI-1, MPI-2) that you are going to install.

You can obtain a summary of the available options with the command ./configure help that will answer: ./configure [OPTION]... [VAR=VALUE]...

To assign environment variables (e.g., CC, CFLAGS...), specify them as VAR=VALUE. See below for descriptions of some of the useful variables. Defaults for the options are specified in brackets.

```
1  Configuration:
2    -h, --help              display this help and exit
3        --help=short        display options specific to this package
4        --help=recursive    display the short help of all the included packages
5    -V, --version           display version information and exit
6    -q, --quiet, --silent   do not print `checking...' messages
7        --cache-file=FILE   cache test results in FILE [disabled]
8    -C, --config-cache      alias for `--cache-file=config.cache'
9    -n, --no-create         do not create output files
10       --srcdir=DIR        find the sources in DIR [configure dir or `..']
11
12 Installation directories:
13   --prefix=PREFIX         install architecture-independent files in PREFIX
14                           [NONE]
15   --exec-prefix=EPREFIX   install architecture-dependent files in EPREFIX
16                           [PREFIX]
17
18 By default, `make install' will install all the files in
19 `NONE/bin', `NONE/lib' etc.  You can specify
20 an installation prefix other than `NONE' using `--prefix',
21 for instance `--prefix=$HOME'.
22
23 For better control, use the options below.
24
25 Fine tuning of the installation directories:
26   --bindir=DIR            user executables [EPREFIX/bin]
27   --sbindir=DIR           system admin executables [EPREFIX/sbin]
28   --libexecdir=DIR        program executables [EPREFIX/libexec]
29   --datadir=DIR           read-only architecture-independent data [PREFIX/share]
30   --sysconfdir=DIR        read-only single-machine data [PREFIX/etc]
31   --sharedstatedir=DIR    modifiable architecture-independent data [PREFIX/com]
32   --localstatedir=DIR     modifiable single-machine data [PREFIX/var]
33   --libdir=DIR            object code libraries [EPREFIX/lib]
34   --includedir=DIR        C header files [PREFIX/include]
35   --oldincludedir=DIR     C header files for non-gcc [/usr/include]
36   --infodir=DIR           info documentation [PREFIX/info]
37   --mandir=DIR            man documentation [PREFIX/man]
38
39 System types:
40   --build=BUILD      configure for building on BUILD [guessed]
41   --host=HOST        cross-compile to build programs to run on HOST [BUILD]
42
43 Optional Features:
44   --disable-FEATURE       do not include FEATURE (same as --enable-FEATURE=no)
45   --enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
46   --enable-64bits         Use 64 bits addressing (default on sgi and fujitsu)
47   --enable-promote-real   Promote REAL fortran data type to DOUBLE PRECISION
48   --enable-blasopti       Use BLAS optimization (default on scalar computers)
49   --enable-mpi_softwait   Use non CPU hogging mpi_wait (default on sgi, sun, nec, linux)
50
51 Optional Packages:
52   --with-PACKAGE[=ARG]    use PACKAGE [ARG=yes]
53   --without-PACKAGE       do not use PACKAGE (same as --with-PACKAGE=no)
54   --without-mpi           Use Monoprocessing without MPI
55   --with-mpich=MPICH_ROOT mpich for MPI (default=no)
56   --with-lam=LAMMPI_ROOT  lam for MPI (default=no)
57   --with-openmpi=OPENMPI_ROOT  OpenMPI for MPI (default=no)
58   --with-mpi_path=path    Path of the MPI implementation
59   --with-F90=F90          F90 compiler
60   --with-CC=CC            C compiler
61   --with-fopt=OPT         Option for Fortran Compiler
62   --with-copt=OPT         Options for C compiler
63   --with-debug=EXTRA_FLAGS    enable debugging (default debug flag  is -g)
64   --with-fortran_underscore   Underscore at end of fortran functions
65   --with-fortran_main=MAIN    internal name of main FORTRAN routine
```

```
66                              (default value depends on system type)
67    --with-roundtrip-delay=roundtrip-delay        *100 MPI_Iprobes (default~100)
68    --with-mpi_comm_free=mpi_comm
69    --with-leak_mem_ctl      To detect memory leak
70    --with-shared_lib       Compile shared libraries
71    --with-mpi1mode           using mpi1 mode (no spawn)
72    --with-mpi2win            using mpi2 windows
73
74  Some influential environment variables:
75    CC          C compiler command
76    CFLAGS      C compiler flags
77    LDFLAGS     linker flags, e.g. -L          if you have libraries in a
78                nonstandard directory
79    CPPFLAGS    C/C++ preprocessor flags, e.g. -I              if you have
80                headers in a nonstandard directory
81    CPP         C preprocessor
```

Use these variables to override the choices made by 'configure' or to help it to find libraries and programs with nonstandard names/locations.

For normal usage, you have to concentrate on the options on lines 13, 15, 40,46 ,47, 54-57, 59-62,65, 70 and 71 only. The remaining options are dedicated to the OpenPALM developers. In any case we suggest to explicitly choose the FORTRAN 90 and C compiler. The PALM Team tested OpenPALM with most of the available compiler suites. Amongst them, notice:

- gcc and gfortran, form the GNU suite
- gcc and g95
- pgcc and pgf90 from the PGI suite PGI
- intel compilers suite
- pathscale compilers suite
- xlc and xlf90 on IBM
- sxmpif90 and sxmpicc on NEC vector supercomputers.

The most thoroughly tested configurations (the ones used at CERFACS) are (pgcc, pgf90) and (gcc, gfortran).

Remark about the GNU compilers (gcc/gfortran) and Intel (icc/ifort): In this case you have to add the option: –with-fortran_main=MAIN__ otherwise it won't be possible to link objects written in C and objects written in FORTRAN.

Important remark: It is absolutely mandatory that the C and FORTRAN compilers are compatible and to use them for compiling (in this given order):

- the MPI library
- the PALM library
- the object libraries for the PALM units
- the PALM applications.

Once you have chosen the compilers, you could maybe have to choose an MPI distribution and indicate it as an option of configure. For the MPI-2 mode, the following public domain distributions have been tested and validated:

- LAM/MPI version 6 and following: option –with-lam=path where LAM/MPI is installed
- OPENMPI version 1.2.7 and following: option –with-openmpi=path where OPENMPI is installed
- MPICH2 version 1.0.7 and following: option –with-mpich=path where MPICH is installed

For the MPI-1 mode, almost every MPI distribution implements the MPI-1 standard with an appropriate quality and completeness.

4.3. *Example of installation on a Mac os X*

For the installation on Mac os X presented here, gfortran and gcc are used as compilers and MPICH2 for the MPI distribution. The compiler gfortran can be downloaded at (http://hpc.sourceforge.net/) and MPICH2 at (http://www.mcs.anl.gov/research/projects/mpich2/)

4.3.1. *Installation of MPICH2*

The installation of MPICH2 is facilitated by the *configure* command:

```
> export FFLAGS=-m64
> ./configure --prefix=$install_dir_mpich2 --exec-prefix=$install_dir_mpich2
--enable-shared
```

Then, the compilation and installation of MPICH2 in the directory *$install_dir_mpich2* is obtained by:

```
> make
> make install
```

4.3.2. *Installation of OpenPALM in the MPI-1 mode*

It is possible to install the OpenPALM library in MPI-1 and MPI-2 modes on the same machine. The configuration commande for the MPI-1 mode reads:

```
> export CFLAGS="-dynamiclib -framework Accelerate   -flat_namespace  -Wl,-undefined,
suppress"
> ./configure --enable-64bits --with-mpich=$install_dir_mpich2 --with-fortran_main=MAIN__
--with-F90=mpif77 -with-CC=mpicc --with-shared_lib
```

At the end of the end of the execution of the *configure*, you see that the OpenPALM library will be installed in

```
PREFIX=$PALM_MP/unknown64r4native_mpi1mode
EXEC_PREFIX=$/PALM_MP/unknown64r4native_mpi1mode
```

where *unknown* stands for the architecture and *native_mpi1mode* for the MPI distribution in MPI-1 mode.

Important remark: note that a change must be done in the OpenPALM *Makefile*. Indeed, this file actually contains the following line:

```
SO_FLAGS=-shared -fPIC
```

that must be replaced by:

```
SO_FLAGS=-dynamiclib -flat_namespace  -Wl,-undefined,suppress -m64
```

Finally, to compile and install the OpenPALM library, tape

```
> make
> make install
> ranlib -c unknown64r4native_mpi1mode/lib/libpalm.a
> ranlib -c unknown64r4native_mpi1mode/lib/libdrv.a
> ranlib -c unknown64r4native_mpi1mode/lib/libdtm.a
```

The *ranlib* command aims at adding or updating the table of contents of archive libraries. Without this operation on the OpenPALM libraries, the creation of the executables of coupling applications if not possible.

4.3.3. *Installation of OpenPALM in the MPI-2 mode*

To install the OpenPALM library in MPI-2 in the same *$PALM_MP* directory, it is important to clean the previous compilation:

```
> make clean
```

Then, the same procedure as for the MPI-1 mode is followed:

```
> export CFLAGS="-dynamiclib -framework Accelerate   -flat_namespace  -Wl,-undefined,
suppress"
> ./configure --enable-64bits --with-mpich=$install_dir_mpich2 --with-fortran_main=MAIN__
--with-F90=mpif77 -with-CC=mpicc --with-shared_lib
```

At the end of the end of the execution of the *configure*, you see that the OpenPALM library will be installed in

```
PREFIX=$PALM_MP/unknown64r4native
EXEC_PREFIX=$/PALM_MP/unknown64r4native
```

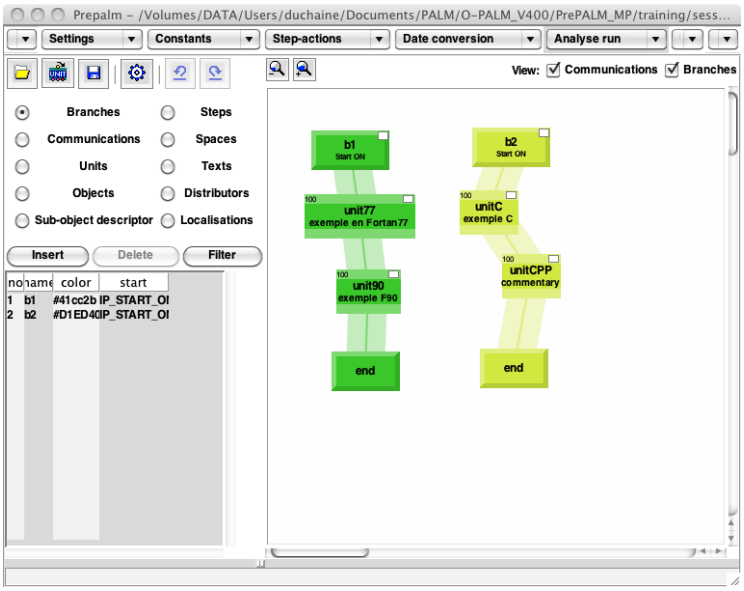

FIGURE 3. Example of verification test with session 2 of the training.

where *unknown* stands for the architecture and *native* for the MPI distribution.

Important remark: note that, as for the MPI-1 installation, a change must be done in the OpenPALM *Makefile*. Indeed, this file actually contains the following line:

```
SO_FLAGS=-shared -fPIC
```

that must be replaced by:

```
SO_FLAGS=-dynamiclib -flat_namespace  -Wl,-undefined,suppress -m64
```

Finally, to compile and install the OpenPALM library, tape

```
> make
> make install
> ranlib -c unknown64r4native/lib/libpalm.a
> ranlib -c unknown64r4native/lib/libdrv.a
> ranlib -c unknown64r4native/lib/libdtm.a
```

As explained for the MPI-1 installation, the *ranlib* command aims at adding or updating the table of contents of archive libraries. Without this operation on the OpenPALM libraries, the creation of the executables of coupling applications if not possible.

4.4. *Checking of the installation and additional informations*

In order to check the installation of OpenPALM in both MPI modes, we use session 2 of the training program (Fig. 3). For more details on this session as well as the use of PrePALM to generate MPI-1 and MPI-2 application, refer to the OpenPALM manual.

The compilation options contained in the *Make.include* are suitable for both the MPI-1 and MPI-2 modes. Note that as *gfortran* is used here, it is necessary to add the reference to *gfortran* library *-lgfortran* in the C flags.

```
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ #
PALMHOME = $PALM_MP/unknown64r4native_mpi1mode

F90 = mpif77
F90FLAGS = -O2 -fconvert=big-endian
LF90FLAGS =
F90EXTLIB =

F77 = mpif77
F77FLAGS = -O2 -fconvert=big-endian
LF77FLAGS =
```

```
F77EXLIB =

FPPFLAGS = -cpp

CC = mpicc
CCFLAGS = -dynamiclib -Wl,-undefined,suppress -flat_namespace
LCCFLAGS =
CCEXTLIB = -L/usr/local/lib/x86_64/ -lgfortran

C++ = mpicxx
C++FLAGS =
LC++FLAGS =
C++EXTLIB = -L/usr/local/lib/x86_64/ -lgfortran

OMPFLAGS =

SOFLAGS = -dynamiclib -Wl,-undefined,suppress -flat_namespace -m64

INCLUDES = -I../../SRC/HEADERS/
LIBS =

USERINCF =

# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ #
```

Once the compilation of the application is done, the MPI-1 application is executed with

```
> mpiexec -np 1 ./palm_main : -np 1 ./main_unit77 :
-np 1 ./main_unit90 : -np 1 ./main_unitC : -np 1 ./main_unitCPP
```

while the MPI-2 application is launched by

```
> mpiexec -np 1 ./palm_main
```