Configuration of the TELEMAC system under the Python environment

links from Installation notes and Python for TELEMAC

Environment Variables

A (small) number of environment variables must be set in your user profile. Namely you have to modify the variable **$PATH** and to create a new variable called **$SYSTELCFG**. Once these are setup, a configuration file (commonly named the *"systel.cfg"* file) has to be modified depending on your operating system configuration.

$PATH

For the installation procedure with Python, the *"~root\scripts\python27"* directory should be added to your **PATH**. Here is the instruction to do so depending on your operating system:

- On Linux and Unix operating system, you can add the following line to your *.profile* or *.bashrc* files, depending on your preferences and on the shell script used:

```
export PATH=/home/user/opentelemac/v7p1/scripts/python27/:$PATH
```

Please note that the symbol **":"** is used as separator on Linux and Unix operating systems.

- On MS Windows operating systems, you need to open your Control Panel's System windows, access your Environment Variables from the Advanced tab and then click New/Edit in the top "user" part for the **PATH** variable to add:

```
c:\opentelemac\v7p1\scripts\python27
```

Please note that the symbol **";"** is used as separator on MS Windows operating systems.

Optionally, if you are using a Fortran Compiler, you should make sure that it is accessible to you within a command line windows or terminal, and if you are using the TELEMAC-MASCARET system in parallel, you should make sure that MPI utilities such as *mpiexec* and *mpif90* depending on operating system are also within your **PATH**.

$SYSTELCFG

The variable **SYSTELCFG** provides TELEMAC-MASCARET with the path to one of the configuration files *"~root/configs/*.cfg"* present in the directory *"~root/configs/"*. In the same way it was done for the **PATH**, here is the instruction to do so depending on your operating system:

- On Linux and Unix operating system, you can add the following line to your *.profile* or *.bashrc* files, depending on your preferences and on the shell script used:

```
export
```

```
SYSTELCFG=/home/user/opentelemac/v7p1/configs/systel.cis-ubuntu.cfg
```

Please note that **SYSTELCFG** only supports one configuration file, altough one configuration file can include multiple configurations.

- On MS Windows operating systems, you need to open your Control Panel's System windows, access your Environment Variables from the Advanced tab and then click New/Edit in the top “user” part for the **SYSTELCFG** variable to add:

```
c:\opentelemac\v7p1\configs\systel.cis-winxp.cfg
```

Again, please note that **SYSTELCFG** only supports one configuration file, altough one configuration file can include multiple configurations.

The file $SYSTELCFG/systel.cfg.

For the installation procedure with Python, the core of the TELEMAC configuration lies in the *“systel.cfg”* file (the name of the file can be defined by the user) situated at the address given by the environment variable **$SYSTELCFG** (see above). This file is directly interpreted by TELEMAC.

The file *“systel.cfg”* is made of one short section defining the list of configuration names of your computer followed by as many sections as you have configurations on your computer. Sections are separated by names in square brackets (i.e. `[Configurations]`) and where each section contains a number of pairs key = value, or key : value (i.e. `configs : fedgfortrans fedgfopenmpi ubugfortrans ubugfopenmpi susgfortrans susgfopenmpi`). The sections withing the configuration file are as follows:

- `[Configurations]`: sets the list of names of every configurations available on your computer.
- `[user-configuration-name]`: referenced by the value of the key configs in the `[Configurations]` section, this section defines principally the options of your system, including the Fortran compiler, the available external libraries (such as MPI or OpenMPI), the root of your TELEMAC system, this version number, etc.

The details of the keys and values for each sections is as follows:

1. [Configurations]

This section has only one keys.

- **`configs`**: Its value is a list of names of configurations defined by the user for a particular computer. The list is space-delimited. The names are arbitrary and chosen by the user but case-sensitive. A number of examples of configurations can be see in the many configurations files (*”*.cfg”*) available from the SVN repository in the *”~root\configs”* subdirectory.

  For instance, on Windows operating systems:

```
configs : wintelscal wintelmpi wing95scal
```

For instance, on Ubuntu Linux operating systems::

```
configs : ubugfortrans ubugfopenmpi
```

For instance, on OpenSUSE Linux operating systems:

```
configs : susgfortrans susgfopenmpi
```

Again, the names in the lists above are arbitrary and could define configurations for a number of computers within an organisation. The *"systel-edf.cfg"* is such a configuration file, where most of the types of computers used at EDF-LNHE are represented.

2. [user-configuration-name]

This section has a larger number of keys necessary to configure your TELEMAC on your computer. Here `[user-configurtaion-name]` takes the value defined by the user, for instance `[wing95scal]`. That name is also used by TELEMAC to create a subdirectory within the tree structure of each module at the same level as the "source" to store the compiled object, libraries and executables created for that specific configuration. In order to help with this setup a number of pre-defined configurations have been stored in *"systel-*.cfg"*. It is therefore recommended that you make a copy of one of the files provided, rename it *"systel.cfg"* and select the most appropriate configs' value for your own configurations.

In this section of the file or in any of these sections (should you define multiple configurations for your computer) the following keys are required:

= root =

- **root**: name of the directory where the TELEMAC system is, or ~root as defined above.

  For instance on MSWindows operating systems:

  ```
  root : c:\opentelemac\v7p1
  ```

  and on Linux operating systems:

  ```
  root : /home/user/opetelemac/v7p1
  ```

= version =

- **version**: sets the version number for the TELEMAC system. For instance:

  ```
  version : v7p1
  ```

= language =

- **language**: sets the language for the TELEMAC system (value 1 for French; value 2 for English) ... this key is not used yet.

= modules =

- **modules**: defines the name of the modules for which the compilation is carried out. The module names that are not recognised as part of the TELEMAC system are ignored. In v6p1, the modules availables are: telemac2d, telemac3d, sisyphe, artemis, tomowac, stbtel, postel3d and parallel. Three additonal names are however allowed:
  - **system**: which TELEMAC automatically replaces by the list of all available modules present on your computer
  - **clean**: which trigger the action (here compilation) to be carried out as if it was the first time
  - **update**: which trigger the action (here compilation) only to update the parts that are necessary.

    For instance, the compilation of the entire TELEMAC system, re-creating all existing objects, libraries and executables would simply write on MSWindows operating system:

    ```
    [wintel32s]
    root: c:\opentelemac\v7p1
    version: v7p1
    modules: clean system
    ```

: It is recommeneded that you just use the following by default:

```
modules: update system
```

Note that the order of appearance of the names of the module has no bearing on the compilation of the TELEMAC system. In addition, TELEMAC will sort out the dependencies between modules automatically, which means, for instance, that the following will compile TELEMAC-2D only:

```
modules: update telemac2d
```

= options =

- 'options': sets the switches between libraries, for instance parallel vs. paravoid, openmi vs. mpi, mumps, XDMF, etc. For instance, when configuring for parallel use:

```
options : parallel mpi
```

= cmd_* =

- **cmd_***: defines the commands and options of your Fortran compiler. There 3 of them: 'cmd_obj', 'cmd_lib', 'cmd_exe', respectively your compiler's directives to create objects, libraries and executables. These keys are commands, written in full as executed in a terminal windows. Within each command, a few tags are defined, such as '<libname>', which are used by the compiler to replace by the appropriate value.

: For instance, for the Intel Fortran compiler

```
'''cmd_obj''': ifort.exe /c /Ot /iface:cref /iface:nomixed_str_len_arg
/nologo
/names:uppercase /convert:big_endian /extend_source:132 '''<mods>'''
'''<incs>''' '''<f95name>'''
'''cmd_lib''': xilib.exe /nologo /out: '''<libname>''' '''<objs>'''
'''cmd_exe''': xilink.exe /nologo /subsystem:console /stack:536870912
/out:<exename> '''<objs>''' '''<libs>'''
```

: (One should note the absence of space between '/out:' and '<exename>')

: For instance, for the gfortran compiler on Linux operating systems:

```
'''cmd_obj''': gfortran -c -O3 -ffixed-line-length-132
-fconvert=big-endian -frecord-marker=4 '''<mods>''' '''<incs>'''
'''<f95name>'''
'''cmd_lib''': ar cru '''<libname>''' '''<objs>'''
'''cmd_exe''': gfortran -fconvert=big-endian -frecord-marker=4 -v -lm -lz -o
'''<exename>''' '''<objs>''' '''<libs>'''
```

: (One should note the space between '-o' and '<exename>')

= mods, libs, incs = * 'mods', 'libs', 'incs': for the compilation of (parts of) the TELEMAC system, the following keys are added to further fine tune the compiler directive, including linking to external libraries. These keys are built based on two words. The first 4 letters, are incs, libs, or mods and define how the compiler should link to include, library and module files. For instance, on Linux operating system, for the gfortran compiler and the MPICH2 implementation, assuming that libmetis is under /usr/lib while the XDMF library is one level up from the ~root directory of the TELEMAC system:

```
'''mods_all''': -M<config>
'''libs_all''': '''<root>'''\..\lib\XDMF\XdmfUtils.a /usr/lib/libfmpich.so
'''incs_parallel''': -I/usr/include/mpich2
'''libs_parallel''': /usr/lib/metis-4.0/libmetis.a
```

: While the '“_all”' are use commonly accorss all modules, it is possible to also specify specific dependencies, such as '“libs_parallel”', for which the libmetis only applies to the executables of the parallel module (partel in particular).

: At the compilation stage the key '<root>' is replaced by '~root' of the TELEMAC system and '<config>' by the appropriate local directory created for a specific module and configuration (where the objects, libraries and executables are).

= sfx_* = * 'sfx_*': defines the suffixes of each type of files created through the compilation commands. There are 4 of them: 'sfx_lib', 'sfx_obj', 'sfx_mod', 'sfx_exe' and 'sfx_zip'. A library would have a '”.lib”' extension on windows while having a '”.a”' extension on Linux, for

instance. The suffix can also be empty, which is appropriate for executables on Linux, for instance, as opposed to '".exe"' on MS Windows operating system. Note that 'sfx_zip' is used to define what type of ZIPPER will be called, depending on the platform. So '".zip"' will launch a windows-like zip archive while '".gztar"' will launch the standard tarball and gzip tools, if installed.

= options' relatives = * 'parallel', 'mpi', etc.: last but not least, some of the options defined with the key '"options"' are completed with their own keys. These keys further inform how TELEMAC is launched. At this stage, only the '"mpi"' option has such key, but a number of other keys will be defined in the future, for example, to put a simulation in a queue system on a cluster of computer.

: For instance on Linux operating systems with the MPICH2 implementation (where the command 'mpiexec' completes the launch of the TELEMAC modules), the following two keys are added to the configuration:

```
'''mpi_cmdexec''': /usr/bin/mpiexec '''<wdir>''' '''<ncsize>''' '''<host>'''
'''<exename>'''
:mpi_hosts: -localonly
```

: The field '<wdir>' is replaced by the absolute path of the temporary sub-directory where the simulation is launched from. The field '<ncsize>' is read from the keyword of your CAS file (PARALLEL PROCESSORS). The field '<host>' is taken from the key '"mpi_hosts"' is there, or replaced by '"-localonly"' by default. The field '<exename>' is replaced by the name of the temporary executable created by TELEMAC within the temporary sub-directory.

Important note

For the installation procedure with Perl, the environment variable 'SYSTELCFG' sets the directory where TELEMAC will find the file '"systel.ini"'. For the installation procedure with Python, the environment variable 'SYSTELCFG' sets the configuration file and its path, without assuming that the name of the file would be '"systel.ini"'. The name of the configuration can be set by the user and will be here referenced by '"systel.cfg"'.