# Telemac程序安装说明

## 大致了解

telemac主要是用F90语法写的（中间夹杂一些F77语法）。大致地了解了一下telemac模拟系统，包括以下几部分：

* ARTEMIS是波浪模型；
* TOMAWAC也是波浪模型；telemac2d耦合的是TOMAWAC
* DREDGESIM是模拟挖沙的（考虑点源的泥沙填入或取走）；
* POSTEL3D是后处理3D计算结果的；
* SISYPHE是泥沙输移模块；
* STBTEL是文件控制模块（负责文件读取等操作的）；
* UTILS放置程序运行需要的一些工具子程序，包括：

BIEF: Bibliotheque d'Elements Finis (Library of Finite Element)

STBTEL:

PARTEL: 首先对TELEMAC2D的网格地形并行分区

telemac是开源代码的程序，一般在Linux系统下使用makefile编译（可以进入进出文件夹，执行相应的编译命令），这种程序一般很难用Visual Studio编译，因为Windows下的代码一般都放一块，难以管理。

## 网上的安装步骤

从网上了解到，Automatic installer是为了新手快速安装telemac的程序，在Windows系统下运行安装程序，会自动安装下面的一系列程序（如python, gfortran, matplotlib等）。

Automatic installer for the Windows platform (64-bit).

The executable will automatically:

* Install Python 2.7;
* Install Python libraries (numpy, scipy, matplotlib);
* Install a SubVersion client;
* Install a FORTRAN compiler (gfortran);
* Configure all the paths required, and create user configuration file;
* Download the openTELEMAC code from the repository (two latest versions);
* Compile the system.
* By default, all components are selected unless the installer detects the component (or an equivalent) is already present.

另外，还有telemac的安装源代码（Windows和Linux的），还有已编译好的可执行程序和中间生成的obj文件。

网上可下载的source\_windows\_xp是v6p3版本的telemac；

可自己下载代码（最新的v7p3），尝试使用Visual Studio 2008 + Intel Visual Fortran 2011编译telemac2d

一般非结构网格有限元的程序，并行计算，都需要metis库对网格进行区域分区(domain decomposition)，telemac网站上可下载metis5的lib文件(ifort编译的)

32 and 64 bits version of METIS 5 library for windows, compiled with Intel Fortran11. Sources can be obtained on the [Karypis Lab](http://glaros.dtc.umn.edu/gkhome/metis/metis/overview) website.

## 代码分析及编译过程

以telemac2d为例：

主程序：homere\_telemac2d.f

各模块下，程序名为：homere\_\*.f为该模块的主程序，起调用其他子程序的作用。

编译telemac模型最方便的方式是，使用python工具在DOS系统下编译（telemac没有使用预处理命令的方式，一次需要编译所有代码，很费时间，据网上信息，编译一次要花半小时）；

静态链接库文件：

libBIEF.lib libdamocles.lib libparallel.lib libsisphe.lib

## Visual Studio 2008 + Intel Fortran 2011编译

需要Visual Studio窗口那种，只能分模块编译

（1）首先，将utils模块编译为静态链接库（debug格式编译一次约5分钟；release格式编译一次约8分钟），vs工程文件在v6p3\sources\utils\utils\_libs

utils程序是一些辅助telemac运行的程序，一般不用关心其原理。

（如果需要使用并行计算功能，需要打开预处理命令：HAVE\_MPI）

（2）telemac2d耦合了泥沙模型sisyphe、海水状态模型tomawac

使用utils的lib文件编译， 可更方便看代码，不需要把所有的代码都添加到工程中。

telemac3d等程序编译，同样如此。

## 编译器参数设置

我注意到telemac编译时ifort的参数设置：(makefile)

ifort.exe /c /Ot /iface:cref /iface:nomixed\_str\_len\_arg /nologo /fpp /names:uppercase /convert:big\_endian <mods> <incs> <f95name>

对应地，要在Visual Studio (IDE)中设置：

Project->Property pages-> Fortran

Preprocessor -> Preprocess Source File -> Yes

Additional Include Directories:添加需要连接的库的路径

Diagnostics \_> Check Routine Interfaces -> No

/c /Ot 优化设置

分2块设置：Fortran和linker

Project->Property pages-> Fortran

General -> Enable Incremental Linking -> No

General -> Suppress Startup Banner -> Yes (/Nologo)

External Procedures > Calling Convention：/iface: cred

External Procedures -> String Length Argument Passing  (/iface:[no]mixed\_str\_len\_arg) 这一条不用设置了，用默认的就行

Name Case Interpretation -> Upper Case

Compatibility > Unformatted File Conversion (设置convert: big\_endian)

Project->Property pages-> Linker

Additional Library Directories:添加需要调用的静态链接库文件的路径

Input: 填写需要调用的静态链接库文件名

### 串行

首先，要用python生成telemac2d运行需要的配置文件：

python C:/path/runcode.py telemac2d -s yourcase.cas

然后运行：telemac2d

串行的运行效率很低。

### 并行

并行计算前，需要做网格分区前处理：PARTEL.F、PARTEL\_PARA和PARTEL\_prelim.F三个主程序；partel是METIS网格分区的程序，可单独编译为三个可执行程序，进行网格分区。

编译PARTEL,VS配置：

INCLUDE:D:\opentelemac\v6p3\sources\utils\bief\libBIEF\Debug;D:\opentelemac\v6p3\sources\utils\parallel\libparallel\Debug

LIB: D:\opentelemac\metis-5\_lib libmetis32b.lib

前处理步骤：

执行partel\_prelim，生成RESULT\_SEQ\_METIS

执行partel（串行METIS）、partel\_para（并行ParMETIS），生成各分区网格文件

步骤：

（1）把opentelemac设置为环境变量，然后注销电脑，使其生效；（作用：可在任意路径下执行程序）；

（2）在cmd下运行：partel (把模拟区域分区，如分成4个区域--对应4个进程）；partel的输入文件为：PARTEL.PAR

其他的输入文件跟串行的一样；

（3）运行并行的telemac2d; 需要一个输入文件：PARAL

配置好MPICH2的并行计算环境；

cmd命令： mpiexec -np 4 telemac2d

（4）运行完了，会生成4个RES文件（计算结果）；

（5）4个分区的计算结果，需要用gretel合并，用于可视化。

## 运行(Python for Telemac)

### Configuration:

config.py -help.

* -c CONFIGNAME (alternatively –configname=CONFIGNAME)
* -f CONFIGFILE (alternatively –configfile=CONFIGFILE)
* -r ROOTDIR (alternatively –rootdir=ROOTDIR)
* –clean

### Compilation:

compileTELEMAC.py

Arguments available under compileTELEMAC.py are listed with compileTELEMAC.py -help.   
They are :

* -c CONFIGNAME (alternatively –configname=CONFIGNAME)
* -f CONFIGFILE (alternatively –configfile=CONFIGFILE)
* -r ROOTDIR (alternatively –rootdir=ROOTDIR)
* -m MODULES (alternatively –modules=MODULES)
* -b (alternatively –bypass)
* –rescan
* –clean

### Running:

runcode.py

Arguments available under runcode.py are listed with runcode.py -help.   
They are :

* -c CONFIGNAME (alternatively –configname=CONFIGNAME)
* -f CONFIGFILE (alternatively –configfile=CONFIGFILE)
* -r ROOTDIR (alternatively –rootdir=ROOTDIR)
* -s (alternatively –sortiefile)
* -t (alternatively –tmpdirectory)
* -x (alternatively –compileonly)
* -w WDIR (alternatively –workdirectory=WDIR)
* –jobname=JOBNAME
* –queue=HPC\_QUEUE
* –walltime=WALLTIME
* –email=EMAIL
* –hosts=HOSTS
* –ncsize=NCSIZE
* –nctile=NCTILE
* –ncnode=NCNODE
* –sequential
* –mpi
* –split
* –merge
* –run
* –use-link

example:

**python runcode.py telemac2d -s t2d\_test.cas –ncsize 36 –nctile 3 -c hydry –queue highp –jobname test -email=Joe.Bloggs@hrwallingford.com**

**python D:/opentelemac/v6p3/scripts/python27/runcode.py telemac2d -s t2d\_break.cas --ncsize 4**

**python D:/opentelemac/v6p3/scripts/python27/runcode.py telemac2d -s t2d\_bridge.cas --ncsize 4**

**集群上的运行：**

**python D:/opentelemac/v7p3/scripts/python27/runcode.py telemac2d -s t2d\_break.cas --ncsize 4 --hosts LIJIAN**

**python D:/opentelemac/v7p3/scripts/python27/runcode.py telemac3d -s t3d\_amr.cas --ncsize 4**

### Validation

validateTELEMAC.py

Arguments available under validateTELEMAC.py are listed with validateTELEMAC.py -help.   
They are :

* -c CONFIGNAME (alternatively –configname=CONFIGNAME)
* -f CONFIGFILE (alternatively –configfile=CONFIGFILE)
* -r ROOTDIR (alternatively –rootdir=ROOTDIR)
* -v VERSION (alternatively –version=VERSION)
* -m MODULES (alternatively –modules=MODULES)
* -s (alternatively –screen)
* -w WDIR (alternatively –workdirectory=WDIR)
* –jobname=JOBNAME
* –queue=HPC\_QUEUE
* –walltime=WALLTIME
* –email=EMAIL
* –hosts=HOSTS
* –ncsize=NCSIZE
* –nctile=NCTILE
* –ncnode=NCNODE
* –mpi
* –split
* –merge
* –run
* -b (alternatively –bypass)
* -k RANK (alternatively –rank=RANK)
* –act=ACTION
* –draw=DRAWING
* –save=SAVING
* –cast=CASTING
* –report=REPORT
* –clean
* –use-link

-c allows the user to specify the configuration to be used from the configurations given in the configuration file. Failing that, if the configuration file does not refer to a specific configuration, one will be selected arbitrarily.

-f allows the user to specify a different configuration file (./configs/systel.cfg or the file given by the environment variable SYSTELCFG are considered by default). The full name of the file (i.e. including path) is then required.

-r allows the user to specify a different root directory (generally corresponding to a different version of the TELEMAC-MASCARET system).

-v specifies the report version number for the validation.

-m gives the list of modules to be validated. Default list is taken from the configuration file modules entry.

-s indicates whether to display on screen or save silently.

-w allows the user to specify the temporary directory to run into. This saves splitting the input files again when the same number of processors is used in a parallel run.

–jobname assigns a jobname to help track the run on HPC.

–queue specifies the HPC queue where the job is to be run.

–walltime specifies a walltime (real time, not CPU time) for the job corresponding to the maximum time it should take. If this limit is exceeded, the HPC queue manager will stop the job.

–email defines the list of users (email addresses) to which the HPC queue manager has to send mail, if any, about the job.

–hosts if specified, the job will only run on one or several of the hosts. ';' delimited

–ncsize gives the number of processors to be used for the job.

–nctile gives the number of nodes to be used for the job.

–ncnode specifies the number of cores to be used on each node (cannot be greater than the number of cores in a node). This can be useful if memory requirements for the job are high.

–mpi (**internal command**).

–split if specified, will only prepare and split the input files but will not run the job.

–merge if specified, will merge the different contributions to the job output into one result file. –merge usually follows a –run command.

–run if specified, will only run the multi-processor job but will not recombine the output into one result file at the end of the job. –run usually follows a –split command.

-b bypasses failures and try to carry out validation to the end. A final report will highlight problem areas that need addressing.

-k 4 prime numbers joined by a '.' identifying the rank of a test case for compilation/running/validation/other. Only test cases with a certain rank will be run through the automatic validation. Default is 0.0.0.0, meaning all test cases.

–act targets specific actions from the XML file (e.g. translate, run, compare princi) and will only perform these.

–draw targets specific plots from the XML file and will only produce these.

–save targets specific data extractions from the XML file and will only perform these.

–cast filters specific casting actions from the XML file and will only perform these.

–report creates a report summary of the validation exercise in csv format.

–clean removes all object files and executables, and result files from the subforlders corresponding to the specified configurations and modules.

–use-link allows the user to setup symbolic links to the input files rather than making copies of the files in the temporary working folder. Available on linux platforms only.

It is possible to use combinations of the above. For example:

**python D:/opentelemac/v7p3/scripts/python27/validateTELEMAC.py**

## telemac2d代码结构分析

主程序：homere\_telemac2d.f

call BIEF\_INIT() ! 初始化telemac2d

call LECDON\_TELEMAC2D() ! 读取控制文件

call BIEF\_OPEN\_FILES() ! 打开telemac2d的相关文件

bief\_init -> read\_config() ! 读取CONFIG文件

LECDON\_TELEMAC2D -> {

call DAMOCLE ! 读取T2DDICO, T2DCAS (dictionary and steering files)

call read\_submit() ! 读取提交作业的参数

}

## 输入输出文件

CONFIG文件：语言(2), 输出文件逻辑号(6) <- read\_config.F

T2DCAS：steering file

T2DDICO: dictionary file

边界条件文件：T2DCLI

地形文件：T2DGEO

运行telemac2d程序后的计算结果文件是：T2DRES

## 后处理

telemac输出文件格式一般为selafin格式，可通过插件，导入tecplot或QGIS软件进行可视化。

（1）提取某网格节点处的计算值。

【1】Yes this possible through the keywords LIST OF POINTS=32939 and NAMES OF POINTS (optional). These keywords will trigger the printout of the wanted graphical variables of this node in the listing. If you want to put the print in a txt file, you have to include subroutine preres\_telemac2d.f in your fortran user file and change LU in lines 392 and 394 with a logical unit (199 for instance). In this case an ascii file called fort.199 will be written in the temporary working folder.

【2】I have a different way of doing the same thing. See script extract\_pt.py available in [pputils](https://github.com/pprodano/pputils). It works something like: pputils

extract\_pt.py -i in.slf -x 100.0 -y 200.0 -o out.txt

where -i is the \*.slf result file, -x and -y are coordinates, and -o is the output file. It gives time series output (for all variables) of the node closest to the one given in the arguments. It's not the fastest or the most efficient script, but it should work.

## 问题解决

（1）运行v7p3版本的WAQTEL模型时，在并行环境下运行，出现：

PARALLEL::ORG\_CHARAC\_TYPE1:

MEMORY PROBLEM WITH THIS COMPILER:

解决：在ORG\_CHARAC\_TYPE1.F中将STOP注释掉，运行可正常完成。

（2）运行validateTELEMAC.py出现问题：\_tkinter 找不到dll

安装了64位的python，就要使用64位的matplotlib

(3)用Visual Studio编译telemac2d时，一定要convert/BIG\_ENDIAN（大端序）