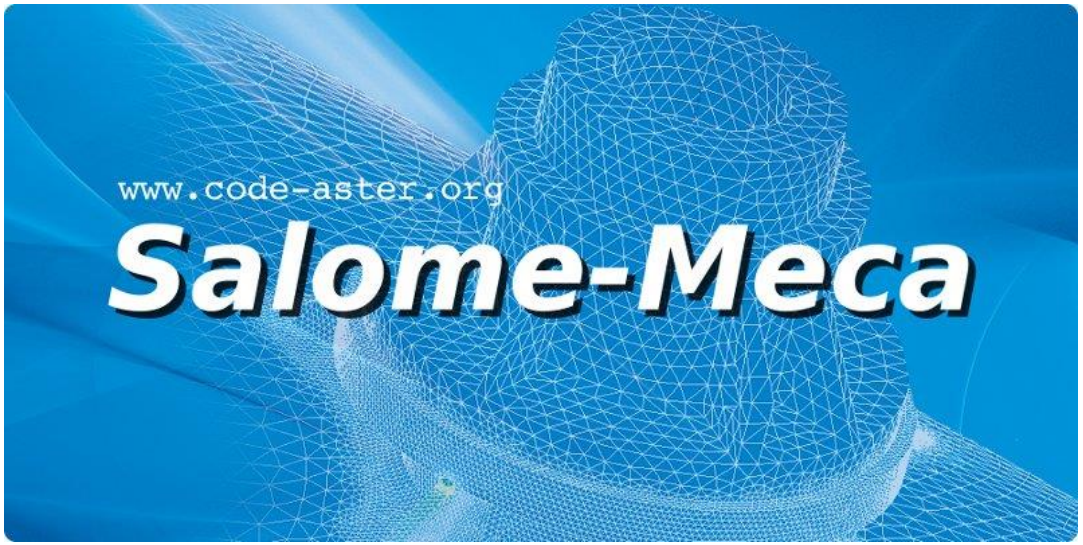
Development in code_aster
ASTERXX in a nutshell



Code_Aster, Salome-Meca course material

Table of contents

asterxx : code_aster redesigned

code_aster, a legacy code

tremendous heritage value
developments becoming tricky
Management of parallelism giving headache

New architecture more suitable for

User interaction
Use of the interactive Python console
Code coupling
Use of Python
Code reuseability
Object Oriented Design
High performance computing
Data encapsulation

What remains the same

Modelling

Old-style syntax still usable

```
DEBUT()

MAIL=LIRE_MAILLAGE(FORMAT='MED',);

MA=DEFI_MATERIAU(ELAS=_F(E=200000.0,
                         NU=0.3,),);

MODE=AFFE_MODELE(MAILLAGE=MAIL,
                 AFFE=_F(TOUT='OUI',
                         PHENOMENE='MECANIQUE',
                         MODELISATION='C_PLAN',),);

MAIL=MODI_MAILLAGE(reuse =MAIL,
                   MAILLAGE=MAIL,
                   ORIE_PEAU_2D=_F(GROUP_MA='haut',),);

MATE=AFFE_MATERIAU(MAILLAGE=MAIL,
                   AFFE=_F(TOUT='OUI',
                           MATER=MA,),);

CHAR=AFFE_CHAR_MECA(MODELE=MODE,
                    DDL_IMPO=(_F(GROUP_MA='bas',
                                 DY=0.0,),
                              _F(GROUP_MA='gauche',
                                 DX=0.0,),),
                    PRES_REP=_F(GROUP_MA='haut',
                                PRES=-100.0,),);

RESU=MECA_STATIQUE(MODELE=MODE,
                   CHAM_MATER=MATE,
                   EXCIT=_F(CHARGE=CHAR,),);

FIN();
```

What does change

Pythonic approach

Position the environment in a shell

```
export PREFIX=/home/myNNI/dev/codeaster/install/
. $PREFIX/share/aster/profile.sh
```

Run Python

```
import code_aster
from code_aster.Commands import *
code_aster.init()

monMaillage = code_aster.Mesh()
monMaillage.readMedFile( 'test001f.mmed' )

monMaillage.debugPrint()
```

Introspection

Try `help(code_aster)` or `help(code_aster.Mesh)`

What does change

Command file example

```
#!/usr/bin/python
# coding: utf-8

import code_aster
from code_aster.Commands import *

code_aster.init()

test = code_aster.TestCase()

monMaillage = code_aster.Mesh()
monMaillage.readMedFile( 'test001f.mmed' )

monModel = code_aster.Model()
monModel.setSupportMesh( monMaillage )
monModel.addModelingOnAllMesh( code_aster.Physics.Mechanics,
                               code_aster.Modelings.Tridimensional )
monModel.build()

acier = DEFI_MATERIAU(ELAS=_F(E=200000.0,
                              NU=0.3,),)

chmat = AFFE_MATERIAU(MAILLAGE=monMaillage,
                      AFFE=_F(TOUT='OUI',
                              MATER=acier,),)


charmeca1 = AFFE_CHAR_MECA(MODELE=monModel,
                           DDL_IMPO=_F(GROUP_MA="Bas",DX=0.,DY=0.,DZ=0.))

imposedPres1 = code_aster.PressureDouble()
imposedPres1.setValue( code_aster.PhysicalQuantityComponent.Pres, 1000. )
charmeca2 = code_aster.DistributedPressureDouble(monModel)
charmeca2.setValue( imposedPres1, "Haut" )
charmeca2.build()

monSolver = code_aster.MumpsSolver( code_aster.Renumbering.Metis )

mecaStatique = code_aster.StaticMechanicalSolver(monModel, chmat)
mecaStatique.addMechanicalLoad( charmeca1 )
mecaStatique.addMechanicalLoad( charmeca2 )
mecaStatique.setLinearSolver( monSolver )

resu = mecaStatique.execute()
resu.debugPrint( )
```

What does change

Supervisor and naming

No more computation supervisor

code_aster objects are real Python pointers

Can be destroyed, stored in lists, dictionaries or objects

Reuse a concept name was forbidden, now it deletes the previous value

Tip : Think "`a=1; print a; a=2; print a`"

tip

JEVEUX objects are no longer named with the user name (automatic named)

After "`MAIL=LIRE_MAILLAGE()`", the corresponding JEVEUX objects and their attributes are no longer « `MAIL .COORDO` » « `MAIL .GROUPMA` » but « `000002.COORDO` » « `000002.GROUPMA` »

confusing for the error messages!

Tip : use "`myObject.debugPrint()`" to see the content of an object

No more 8 characters limitation

Since code_aster objects are pointers, they can be as long as wanted

tip

What does change

Syntax

Old-style and new-style syntax can be mixed

In order to load the commands syntax, one needs "`from code_aster.Commands import *`"

Commands and methods

code_aster objects (concepts) are embedded in C++ objects

The commands are now methods on the objects

"`myMesh=LIRE_MAILLAGE()`" is fully equivalent to "`myMesh=code_aster.Mesh();`
`myMesh.readMedFile('myMEDFile.med')`"

Definition and execution phases

All definition objects have a "`build`" method, all resolution objects have a "`execute`" method

End of presentation

Is something missing or unclear in this document?

Or feeling happy to have read such a clear tutorial?

Please, we welcome any feedbacks about code_aster training materials.

Do not hesitate to share with us your comments on the code_aster forum dedicated thread.