The study

In this case we'll use **Salomé**, **ASTK** and **Code_Aster** to load and combine several separate mesh files into one big mesh and solve for an applied load. Then separate the mesh again, now containing the calculated fields.

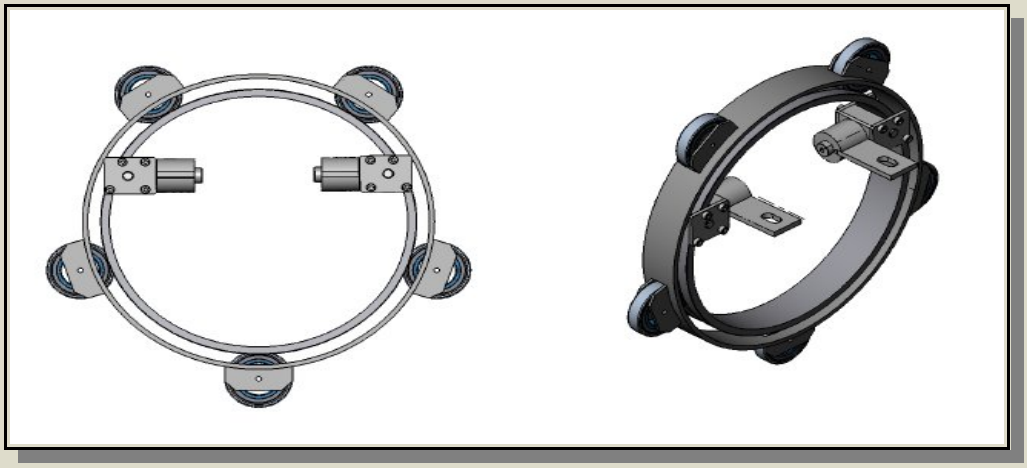
Table of Contents

Introduction and theory

The reasons for this approach can be many, such as Salomé running out of available memory, operations take exceedingly long to complete because the whole mesh has to be displayed/updated etc., or sometimes it's just more practical to work on one part of an assembly instead of the full assembly. This particular study was too much to handle using quadratic elements and having SaloméMECA open at the same time, using my 1Gb laptop. It is not so much an issues using my current workstation.

In Salomés geometry module the full assembly is manipulated in different ways, but each of the parts are meshed and exported separately.

A section of the assembly below, is what we'll be working with.

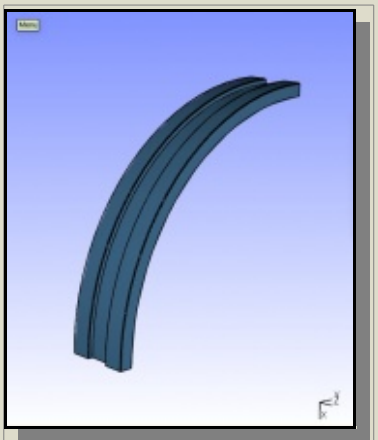
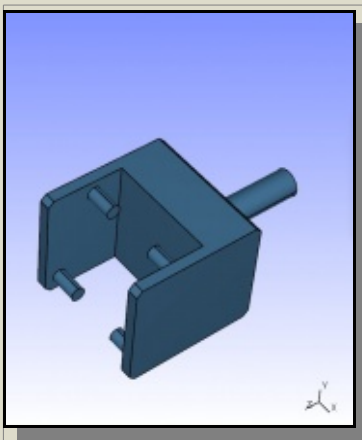
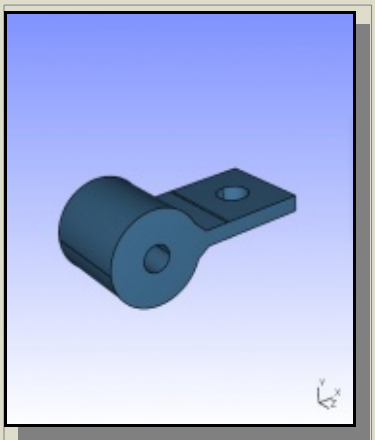
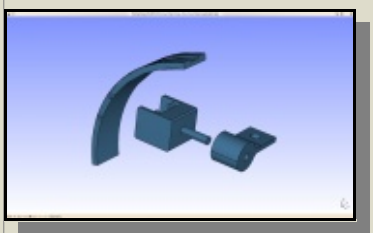
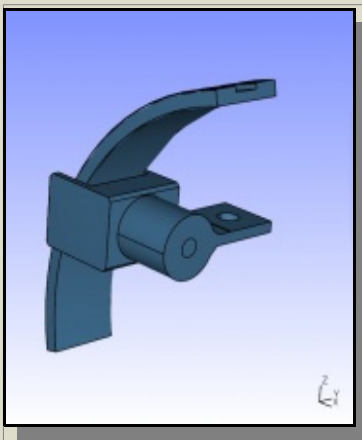


Work flow

To accomplish this feat using **Salomé**, **ASTK** and **Code_Aster**, a few steps must be completed.

- Decide which surfaces of the parts that will be 'glued' together and assign mesh groups accordingly
- Assign unit numbers in **ASTK**
- Tell **Code_Aster** which mesh files to read using unit numbers and tell it which surfaces should be glued together.
- Calculate the results, then print results to each separate mesh

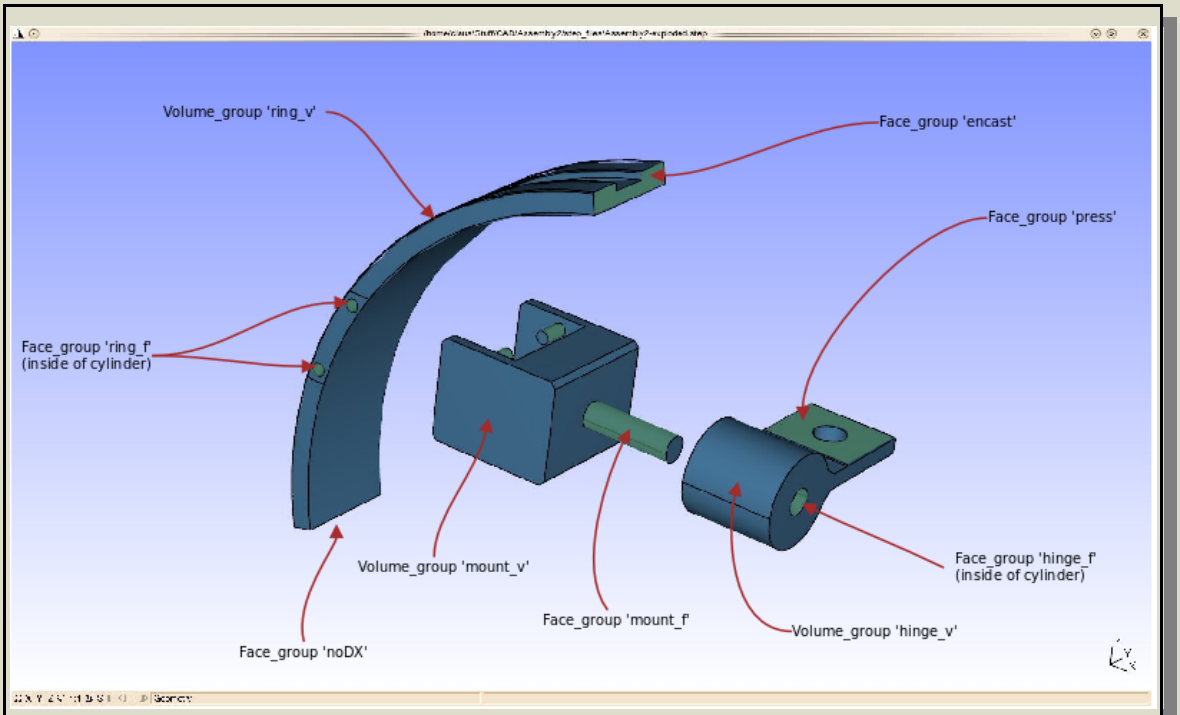
Below in the table the different parts that go into the assembly can be seen.

Tables	Are	Fun
 "Part:Ring"	 "Part:Mount"	 "Part:Hinge"
 "Part:Assembly exploded"	 "Part:Assembly"	

Assignment of groups in Salomé

You should be familiar with assigning groups, meshing and exporting files in Salomé, so I will not go through it here. Consult the **.hdf** file I've attached at the bottom of the page.

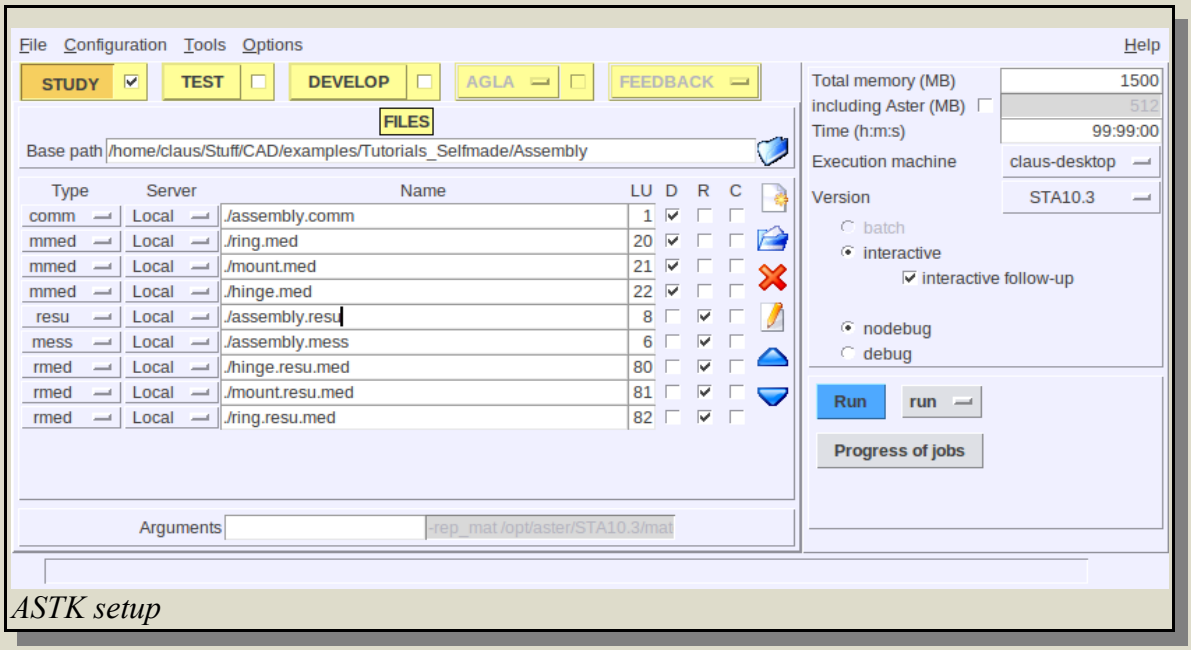
note: I have used Salomé 5.1.5 to generate the mesh, so the **.hdf** might not be backwards compatible. To recreate the mesh, use the included **.brep** files, the diagram below, and use 'automatic tetrahedralization' with a average/local size of 3.



Group diagram

ASTK setup

Each exported mesh file is assigned a unique unit number in **ASTK** so **Code_Aster** can recognize them during parsing of the *.comm* file.



ASTK setup

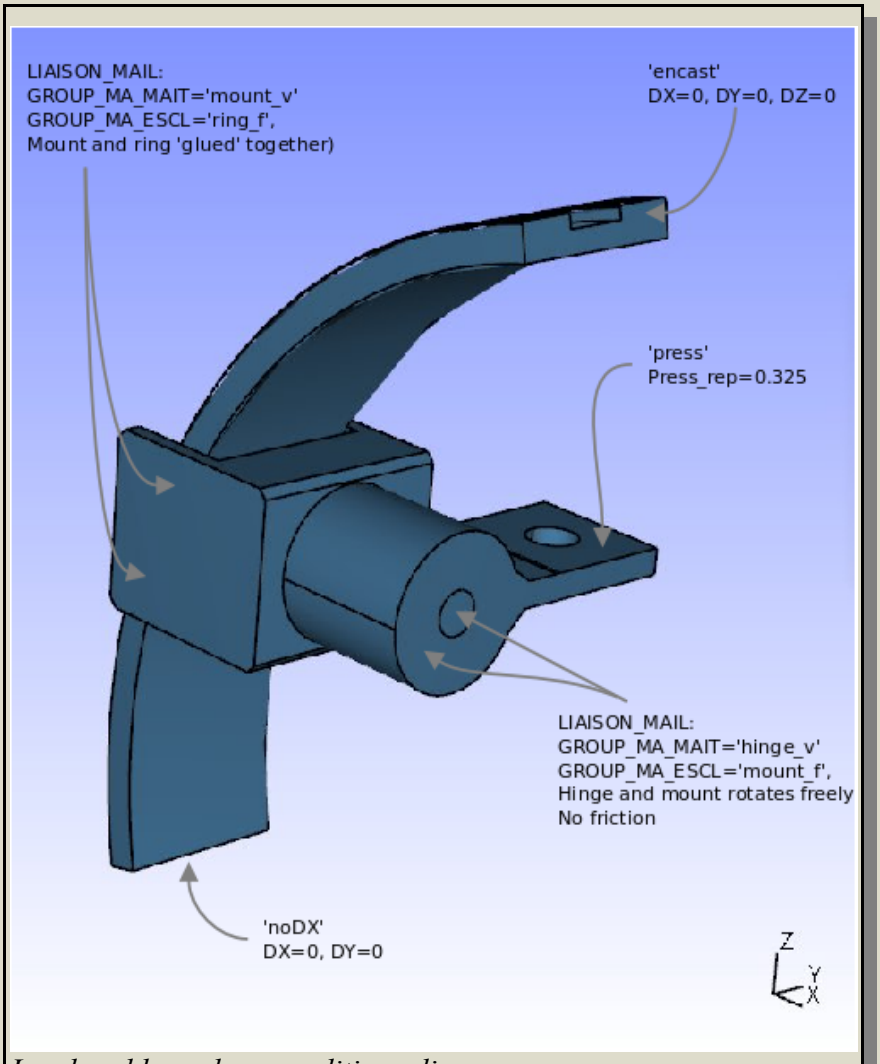
- Assigning the *input* mesh files in **ASTK**
  - *mmed* for mesh file
    - *local* file
      - Name of file on disk
        - *LU*: unique number correspondent to a number in the *.comm* file
          - *D* for Data
- Assigning the *output* mesh files in **ASTK**
  - *rmed* for mesh file
    - etc. etc.

Code_Aster setup

The way Code_Aster connects different meshes, is by using the **LIAISON_MAIL** command (see **U4.44.01** section **4.14**).

A 3D volume group is connected to a 2D face group using a 'master/slave' relationship called *GROUP_MA_MAIT* and *GROUP_MA_ESCL* – this explains why the parts in the group diagram are assigned groups called *name_f* for face and *name_v* for volume.

Heres a diagram of the boundary conditions – They will be explained further in the *.comm* file.



Load and boundary conditions diagram

.comm file, step by step

```
#Claws - March - 2011
#For www.CAELinux.com
#Assembly tutorial

DEBUT();

MA=DEFI_MATERIAU(ELAS=_F(E=2.1e5,
                         NU=0.28,),);
```

Definition

- Self serving credits and date
- DEFI_MATERIAU: Define material, assign the name **MA** to it.
  - ELAS: We only deal with a regular elastic material here, with an elasticity module (Young's module) of 210 GPA and a Poisson's ratio of 0.28

```
ring=LIRE_MAILLAGE(UNITE=20,
                   FORMAT='MED',);

mount=LIRE_MAILLAGE(UNITE=21,
                    FORMAT='MED',);

hinge=LIRE_MAILLAGE(UNITE=22,
                    FORMAT='MED',);
```

Definition

- Read each of the mesh files assigned in **ASTK**
  - UNITE: Uniquely assigned number in **ASTK** (*LU*)

```
mesh1=ASSE_MAILLAGE(MAILLAGE_1=hinge,
                    MAILLAGE_2=mount,
                    OPERATION='SUPERPOSE',);

mesh2=ASSE_MAILLAGE(MAILLAGE_1=mesh1,
                    MAILLAGE_2=ring,
                    OPERATION='SUPERPOSE',);
```

Definition

- ASSE_MAILLAGE - Assemble mesh (See **U4.23.03** for explanation)
  - mesh1: 'Assemble' two mesh files *hinge* and *mount* - use superposition or 'overlay'
  - mesh2: 'Assemble' two mesh files - This time use the *mesh1* previously created and add *ring* to the combined mesh - use superposistion or 'overlay'

```
linmod=AFFE_MODELE(MAILLAGE=mesh2,
                   AFFE=_F(TOUT='OUI',
                           PHENOMENE='MECANIQUE',
                           MODELISATION='3D',),);

mesh2=MODI_MAILLAGE(reuse =mesh2,
                    MAILLAGE=mesh2,
                    ORIE_PEAU_3D=_F(GROUP_MA=('press','hinge_f','mount_f','e
ncast','noDX',),),);
```

Definition

- Assign a 3D mechanical model to everything on *mesh2*
- Reorient the normals of the face groups

```
Qmesh=CREA_MAILLAGE(MAILLAGE=mesh2,
                    LINE_QUAD=_F(TOUT='OUI',),);

qmod=AFFE_MODELE(MAILLAGE=Qmesh,
                 AFFE=_F(TOUT='OUI',
                         PHENOMENE='MECANIQUE',
                         MODELISATION='3D',),);
```

Definition

- Qmesh: Convert the original linear mesh to a quadratic mesh
- qmod: Assign a 3D mechanical model to everything

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

Definition

- Assign the material **MA** to everything, call the field **MATE**

```
CHAR=AFFE_CHAR_MECA(MODELE=qmod,
                    DDL_IMPO=(_F(GROUP_MA='encast',
                                 DX=0.0,
                                 DY=0.0,
                                 DZ=0.0,),
                              _F(GROUP_MA='noDX',
                                 DX=0.0,
                                 DY=0,),),
                    LIAISON_MAIL=(_F(GROUP_MA_MAIT='hinge_v',
                                     GROUP_MA_ESCL='mount_f',
                                     TYPE_RACCORD='MASSIF',),
                                  _F(GROUP_MA_MAIT='mount_v',
                                     GROUP_MA_ESCL='ring_f',
                                     TYPE_RACCORD='MASSIF',),),
                    LIAISON_UNIF=_F(GROUP_MA='press',
                                    DDL='DZ',),
                    PRES_REP=_F(GROUP_MA='press',
                                PRES=0.325,),);
```

Definition

- Assign loads and boundary conditions:
  - Impose zero displacements to face group *encast* and allow *noDX* to only move in the Z direction
  - Use LIAISON_MAIL to 'glue' the *VOLUME* group *hinge_v* to *FACE* group *mount_f* - *hinge_v* can rotate freely around *mount_f*, but not slide off.
  - Use LIAISON_MAIL to 'glue' the *VOLUME* group *mount_v* to *FACE* group *ring_f*
  - Use LIAISON_UNIF to make sure the face group *press* deforms uniformly in the Z direction
  - Apply force to the face group *press*

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

RESU=CALC_ELEM(reuse =RESU,
               MODELE=qmod,
               CHAM_MATER=MATE,
               RESULTAT=RESU,
               OPTION=('SIGM_ELNO_DEPL','EQUI_ELNO_SIGM',),
               EXCIT=_F(CHARGE=CHAR,),);

RESU=CALC_NO(reuse =RESU,
             RESULTAT=RESU,
             OPTION=('SIGM_NOEU_DEPL','EQUI_NOEU_SIGM',),);
```

Definition

- Calculate a solution using the material and loads/boundary conditions
- Calculate the results at the elements
- Calculate the results at the nodes

```
IMPR_RESU(FORMAT='RESULTAT',
          RESU=_F(RESULTAT=RESU,
                  NOM_CHAM='EQUI_ELGA_SIGM',
                  NOM_CMP=('VMIS','TRESCA',),
                  VALE_MAX='OUI',),);
```

Definition

- Extract maximum Von Mises and Tresca stresses (NOM_CMP = component name) from the solution (ELGA = Gauss points) and write the results to the .resu file.
- (Alternatively, the results can be written to a specified file; use 'unit' command for this.)
- This is done before the Gauss points are stripped from the result by the PROJ_CHAMP command.

```
qresu=PROJ_CHAMP(RESULTAT=RESU,
                 MODELE_1=qmod,
                 MODELE_2=linmod,);
```

Definition

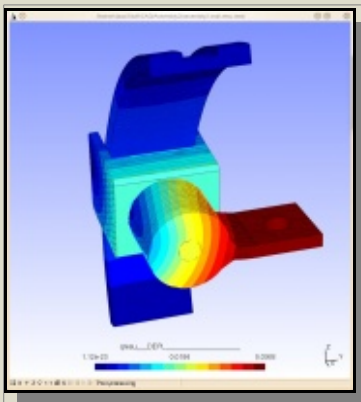
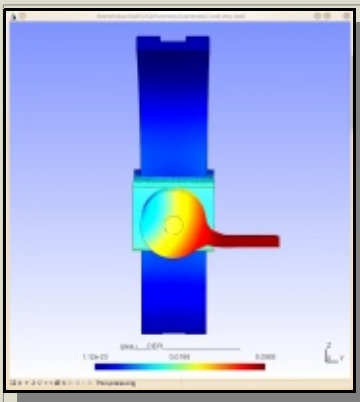
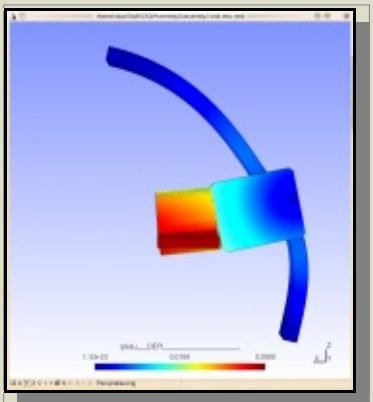
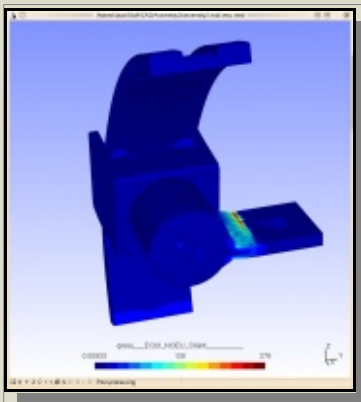
- Project the high definition quadratic model **qmod** optained from **RESU** onto the lower definition linear model **linmod**, call the projected result **qresu**

```
IMPR_RESU(FORMAT='MED',
          RESTREINT=_F(GROUP_MA='hinge_v',),
          RESU=_F(MAILLAGE=hinge,
                  RESULTAT=qresu,
                  NOM_CHAM=('SIGM_NOEU_DEPL','EQUI_ELNO_SIGM','DEPL','SIEF_E
LNO_ELGA',),),);

IMPR_RESU(FORMAT='MED',
          UNITE=81,
          RESTREINT=_F(GROUP_MA='mount_v',),
          RESU=_F(MAILLAGE=mount,
                  RESULTAT=qresu,
                  NOM_CHAM=('SIGM_NOEU_DEPL','SIEF_ELNO_ELGA','EQUI_ELNO_SIG
M','DEPL',),),);

IMPR_RESU(FORMAT='MED',
          UNITE=82,
          RESTREINT=_F(GROUP_MA='ring_v',),
          RESU=_F(MAILLAGE=ring,
                  RESULTAT=qresu,
                  NOM_CHAM=('SIGM_NOEU_DEPL','EQUI_ELNO_SIGM','SIEF_ELNO_ELG
A','DEPL',),),);
FIN();
```

Definition

- Write the results to each individual mesh file, defining which physical file with **UNITE** and *restrain*ing the results to a corresponding volume group

---

Tables	Are	Fun
 "Part:Ring"	 "Part:Mount"	 "Part:Hinge"
 "Part:Assembly exploded"	row 2, cell 2	row 2, cell 3

Notes regarding this study

It is always recommended to extract the nummerical values of the EQUI_ELGA_SIGM field and print it to the **.resu** file, rather than relying on Salomés (or other) graphical representation of e.g. Von Mises stress - this has to do with the way the values are extrapolated from the Gauss points onto the nodes when displaying EQUI_NOUE_SIGM. This extrapolation results in inaccurate results, i.e. too high of a value or even negative values. This has been discussed several times on the Code_Aster forum.

ELGA fields cannot yet be projected back to a linear mesh

Links:

CAELinux

www.caelinux.com

This tutorial with larger images and files from the study attached:

http://www.caelinux.org/wiki/index.php/Contrib:Claws/Code_Aster/10_x_cases/liaison_mail

Code_Aster

www.code-aster.org