SALADYN

EdF Applications.

ANR Project Saladyn
ANR-08-COSI-014

Liverable L3b.2

February 19, 2013

Document d'utilisation du couplage *SALADYN* entre *Code_Aster* et *LMGC90*.

Résumé:

L'objectif principal de ce document est de présenter la mise en œuvre d'une méthode de couplage, accessible via un module *Salomé* appelé *SALADYN*, entre les deux codes de calcul : *Code_Aster* et *LMGC90* pour résoudre les problèmes d'interaction entre milieux continus et milieux discrets. Le premier logiciel est déployé pour modéliser des milieux continus par éléments finis (comme par exemple une fondation), tandis que le second est utilisé pour traiter des interactions dynamiques non linéaires entre solides disjoints (comme par exemple des blocs reposant sur la fondation) ; le couplage sert à établir l'équilibre dynamique non linéaire entre le milieu continu et les milieux divisés.

Présentation du document :

1. Introduction ;
2. Architecture générale du module *SALADYN* ;
3. Installation de *SALADYN* ;
4. Réalisation d'un couplage sur un cas d'application standard.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Table des matières

1 Introduction....3
1.1 Contexte et objectif....3
1.2 Objectif de travail....3
1.3 Remerciements....4
2 Architecture générale du module SALADYN....4
3 Installation du module SALADYN....5
3.1 Contenu du répertoire SALADYN....5
3.1.1 Contenu du répertoire [aster]....6
3.1.2 Contenu du répertoire [LMGC90v2]....6
3.1.3 Contenu du répertoire [prerequis]....6
3.1.4 Contenu du répertoire [SRC]....7
3.2 Compilation des paquets SALADYN....7
3.2.1 Installation du paquet distribué....7
3.2.2 Utilisation du module SALADYN....7
4 Réalisation d'un couplage SALADYN....8
4.1 Mise en données....8
4.2 Préparation des commandes....10
4.2.1 Syntaxe de la macro-commande....10
4.2.2 Exemple : contact de 3 billes déformables....11
4.3 Lancement d'une étude avec couplage....12
4.4 Résultats d'une étude avec couplage....13
5 Bibliographie....13
6 Description des versions du document....14

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

1 Introduction

1.1 Contexte et objectif

Il existe de nombreuses situations physiques où des structures divisées sont en interaction mécanique avec des milieux continus : il peut s'agir de blocs, de colis ou d'équipements simplement posés sur leur support, un enrochement ou milieu granulaire limité par des parois solides et reposant sur une fondation continue... Les interactions entre ces structures divisées et les solides modélisés par des milieux continus déformables sont du type contact frottant, éventuellement intégrant des lois de choc, de cohésion... On envisage autant des situations quasi-statiques que dynamiques. Les lois d'interaction conduisent au caractère non régulier des équations du mouvement, notamment des discontinuités de vitesse et des non-définitions des accélérations, ce qui nécessite une algorithmique particulière, différente selon le type de milieux considéré : divisé ou continu. C'est pourquoi ces fonctionnalités ne sont que rarement disponibles dans le même outil de simulation.

Ce document a pour le but de présenter les grandes étapes de la modélisation de ce type de problèmes et la mise en œuvre au sein de l'environnement *Salomé* du module appelé *SALADYN*, qui utilise sur une même plate-forme :

- *Code_Aster* : ce logiciel libre de simulation numérique en mécanique des structures [3] sera dédié essentiellement à la résolution de problèmes de calcul des structures et thermomécaniques en mécanique des milieux continus déformables non-linéaires ;
- *LMGC90* : développé par le LMGC de Montpellier [4]. Cette plate-forme logiciel libre est construite en s'appuyant sur l'approche *Non Smooth Contact Dynamics* (NSCD) [5]. Elle permet de modéliser des systèmes multi-corps en interaction et de traiter des problèmes de contact, aussi bien entre des corps rigides qu'entre des corps déformables, en 2D et 3D [6] ;
- *Siconos* : développé par l'INRIA-Bipop [7]. Logiciel libre dédié au calcul en dynamique non-régulière.

Le module *SALADYN* permet donc de bénéficier des spécificités des outils comme *Code_Aster* et *LMGC90* dans leurs domaines respectifs, et utilise une méthode de couplage externe modélisant les interactions dynamiques entre milieux continus-divisés.

1.2 Objectif de travail

On souhaite pouvoir utiliser les lois de comportement de matériaux non-linéaires implémentées dans *Code_Aster* pour représenter le comportement des milieux continus déformables des applications en dynamique non-régulière *LMGC90* (ou *Siconos*). Pour cela, il est indispensable de pouvoir piloter étape par étape les opérateurs de calculs non-linéaires de Code_Aster (STAT_NON_LINE [8-9] et DYNA_NON_LINE [10-11]) depuis l'interpréteur python.

Concernant le couplage entre les codes mentionnés ci-dessus, une première maquette python a été réalisée pour coupler *Code_Aster* et *LMGC90*: elle utilisait pour cela les commandes éclatées de DYNA_TRAN_LINE (simulations dynamiques élastiques) [2].

Un travail récent a été effectué par S. Mazet, A. Assire et H-P Cao dans le module de couplage *SALADYN* [12] pour utiliser la version éclatée de la commande STAT_NON_LINE (simulations quasi-statiques non-linéaires). La commande éclatée de DYNA_NON_LINE vient de compléter la maquette de couplage *SALADYN* dans le cadre de la dynamique non-linéaire. C'est pourquoi, dans la suite, on

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

décrit la procédure de couplage entre *Code_Aster* et *LMGC90* pour résoudre des problèmes de contact multi-corps rigides-déformables dans les simulations dynamiques et quasi-statiques.

1.3 Remerciements

Ce document constitue un produit (Tâche 3b.4) du projet ANR *SALADYN* [13], collaboration entre industriels *(EDF R&D, Schneider-Electric)* et laboratoires de recherche *(INRIA/BIBOP, LMGC UMR CNRS, LaMSID UMR CNRS*), ayant bénéficié du soutien de *l'Agence Nationale de la Recherche* sur les années 2009-2012, dans le cadre du programme *COSINUS*.

2 Architecture générale du module SALADYN

L'outil numérique du couplage entre *Code_Aster* et *LMGC90* a été écrit en python et est intégré dans le module *SALADYN*. L'architecture générale du module *SALADYN* est présentée sur la figure (1). Dans ce module, *Code_Aster* et *LMGC90* partagent les mêmes données (géométrie, maillage, comportement du matériau ...), conditions initiales et limites, le même schéma d'intégration en temps adapté à la dynamique non régulière (θ -schéma) ainsi que le même modèle de calcul. *LMGC90* permet de calculer dans un premier temps les forces d'interaction entre les corps en contact. *SALADYN* va ensuite mettre à jour les informations pour *Code_Aster* et *LMGC90*.

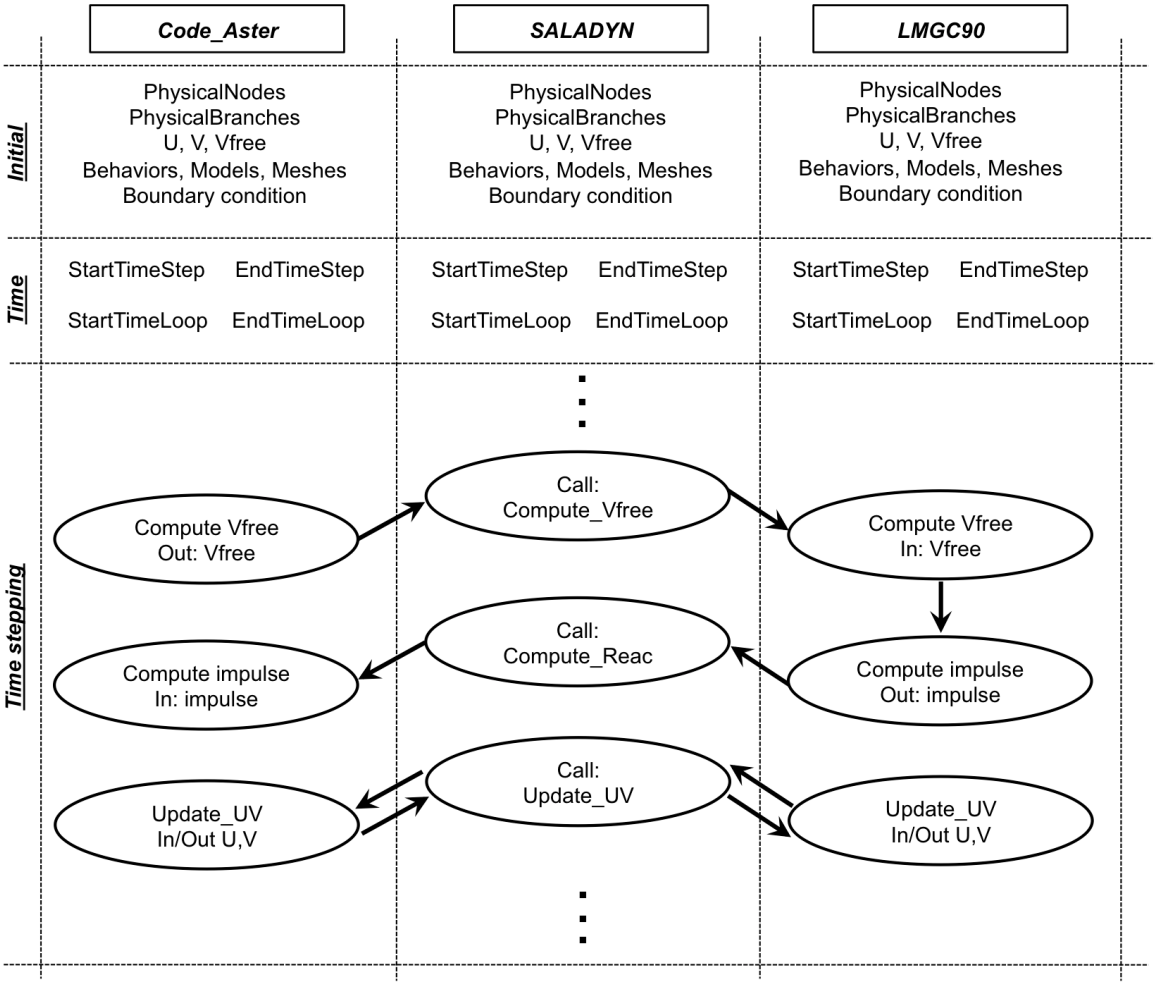


Figure 1: L'architecture générale du module SALADYN.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

3 Installation du module SALADYN

Pour installer le module *SALADYN*, il faut disposer d'un système d'exploitation ***Linux Debian Squeeze 64 bits*** (système à la base de Calibre 7). Tous les paquets concernant le module *SALADYN* sont sauvegardés dans le répertoire *SALADYN* suivant ($ROOT_SALADYN$) **.**

3.1 Contenu du répertoire *SALADYN*

Le contenu inférieur du module *SALADYN* ($ROOT_SALADYN$) est le suivant :

```
$ROOT_SALADYN$/
    ----> [aster]
    ----> [LMGC90v2]
    ----> [prerequis]
    ----> [SRC]
    ----> [Toolbox]
    ----> postinstall.sh
    ----> envSaladyn.sh
```

Ce module contient les deux codes de calcul principaux [Code_Aster] et [LMGC90]. Le répertoire [Toolbox] contient le code de la toolbox du module *SALADYN* pour le couplage entre *Code_Aster* et *LMGC90*. Le répertoire [SRC] contient uniquement les répertoires de sources des logiciels. Enfin, le répertoire [prerequis] contient tous les prérequis nécessaires au fonctionnement du module *SALADYN*. Les contenus de ces répertoires sont détaillés dans la suite.

3.1.1 Contenu du répertoire [aster]

Le répertoire $ROOT_SALADYN$/aster contient une compilation complète de Code_Aster :

```
$ROOT_SALADYN$/aster/
    ----> [bin]        ----> as_run & astk
    ----> [etc]
    ----> intel        ----> ../prerequis/intel/
    ----> [lib]
    ----> [NEW11]
    ----> [outils]
    ----> [postinstall]
    ----> [public]
    ----> [scripts]
    ----> [share]
```

3.1.2 Contenu du répertoire [LMGC90v2]

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Le répertoire $ROOT_SALADYN$/LMGC90v2 contient une compilation complète de *LMGC90*, dans sa version adhoc pour le module [4].

```
$ROOT_SALADYN$/LMGC90v2/
    ----> [LMGC90v2_aa]
    ----> [LMGC90v2_BindingExternalLinearAlgebra]
    ----> [LMGC90v2_BindingExternalModels]
    ----> [LMGC90v2_dev]
```

3.1.3 Contenu du répertoire `[prerequis]`

Le répertoire $ROOT_SALADYN$/prerequis contient des paquets nécessaires pour compiler *SALADYN*.

```
$ROOT_SALADYN$/prerequis/
    ----> [cmake-2.8.6]
    ----> [intel]
    ----> [numpy-1.4.1]
    ----> [old]
    ----> [py-1.3.3]
    ----> [Python-2.6.6]
    ----> [python-graph-1.8.0]
    ----> [python_packages]
    ----> [swig-2.0.4]
```

3.1.4 Contenu du répertoire [SRC]

Le répertoire $root_SALADYN$/SRC contient les sources de *LMGC90*, Toolbox et prerequis.

```
$root_SALADYN$/SRC/
    ----> [LMGC90v2]
    ----> [Toolbox]
    ----> [prerequis]
```

3.2 Compilation des paquets *SALADYN*

Le répertoire *SALADYN* ($root_SALADYN$) contient tout l'environnement de *SALADYN*. Pour la connexion des codes de calcul et des paquets nécessaires, les deux étapes suivantes sont nécessaires.

3.2.1 Installation du paquet distribué

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Le procédure d'installation est la même que pour Salomé-Méca.
Une fois le paquet de distribution SALADYN-vx.x.tgz extrait sur le système de fichier à l'endroit désiré, l'étape suivante, effectuée une seule fois, consiste à configurer le répertoire d'installation :

```
tar zxf SALADYN-vx.x.tgz
cd SALADYN/
    ----> ./postinstall.sh
```

Cette étape est réalisée avec succès lorsqu'il y a le message suivant:

```
Done..
Please source the Saladyn environment :
cd $root_SALADYN$ ; source envSaladyn.sh
```

3.2.2 Utilisation du module *SALADYN*

Le mécanisme de mise en place de l'environnement d'exécution est le même que pour Salomé-Méca. Pour utiliser le module Saladyn, il convient de positionner correctement toutes les variables d'environnement, afin de rendre accessibles les logiciels du paquet *SALADYN*. Pour cela, un script est fourni dans le répertoire d'installation :

```
$SALADYN$root$
    ----> source envSaladyn.sh
```

Cette étape est réalisée avec succès lorsqu'il y a le message suivant :

```
Checking... LD_LIBRARY_PATH

Checking... PATH

Checking... PYTHONPATH
```

Ensuite, pour lancer le couplage, on utilise la procédure habituelle de lancement de *Code_Aster* en passant par as_run par exemple. Il est nécessaire d'utiliser la version surchargée de *Code_Aster* fournie dans le module *SALADYN*, qui donne accès aux macros de couplage MACR_LMGC_LINE et MACR_LMGC_NOLI. Voir ci-dessous pour les détails du fichier .export.

A terme, on vise l'intégration dans Salomé-Méca par le mécanisme de plugin. Le lancement de *SALADYN* se fera en exécutant la commande XXX (script *Python* dans *Salomé*) ou en cliquant sur l'icône XXX dans le menu XXX.

4 Réalisation d'un couplage *SALADYN*

4.1 Mise en données

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Pour réaliser le couplage entre *Code_Aster* et *LMGC90*, il est nécessaire de préparer les mêmes données pour *Code_Aster* et *LMGC90* à fournir dans le module *SALADYN*.

Par exemple, pour une étude placée dans le répertoire XXXX, l'organisation des répertoires pour le couplage est la suivante :

- Pour *Code_Aster*, les données de maillage, de commandes de modélisation, sont sauvegardées dans le répertoire /XXXX :

```
$pwd$/XXXX/
    ----> .med; .msh; .comm; .mail; etc
```

- Pour *LMGC90*, voir la documentation sur [4], les données sont sauvegardées dans le répertoire /XXXX/Data/DATBOX :

```
$pwd$/XXXX/Data/
    ----> /DATBOX/
            ----> BODIES.DAT
            ----> BULK_BEHAV.DAT;
            ----> DRV_DOF.DAT;
            ----> MODELS.DAT
        ...
    ----> /OUTBOX
    ----> /DISPLAY
    ----> /POSTPRO
```

L'utilisateur doit veiller à ce que les données de maillage, de conditions initiales, et de modélisation pour les corps maillés dans le *DATBOX* de *LMGC90* correspondent exactement aux données de *Code_Aster*. Cette étape est très importante car aucune vérification n'est faite au cours du couplage.

Le maillage est conçu de manière unique tant pour *Code_Aster* que pour *LMGC90*. On définira les groupes de mailles nécessaires pour nommer les solides en interaction et les surfaces de contact. Les données de maillage sont répliquées dans le fichier *BODIES.DAT*. En revanche, les corps rigides qui seront considérés dans *LMGC90* sont décrits uniquement dans le fichier BODIES.DAT.

Pour éviter tous problèmes liés à la mise en données pour le couplage, on conseille d'utiliser le format de fichier de maillage .msh dans *Code_Aster.* Le pré-processeur de *LMGC90* sert à convertir les fichiers maillage de type .msh pour l'adapter à *LMGC90 (voir pré-processeur de LMGC90).* En revanche, pour utiliser un fichier .med dans *LMGC90*, il convient de le convertir d'abord au format gmsh, à l'aide du convertisseur de *ASTK* par exemple, puis d'utiliser le pré-processeur *LMGC90* pour verser le maillage dans le fichier *DATBOX*. Ce processus sera automatisé.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

On présente ici le type de fichier maillage.msh obtenu à partir du logiciel *Salomé* pour *Code_Aster*, ainsi que la version *DATBOX* du même maillage.

- Code_Aster (type de fichier .msh) :

```
                    ##### concernant les nœuds #####
  1  1.7999999999999999E-01  1.7500000000000002E-01  0.0000000000000000E+00
  2  2.6500000000000001E-01  9.0000000000000011E-02  0.0000000000000000E+00

                    ##### concernant les éléments triangles #####
  113  2    10000    10000       3     54     19     30
  114  2    10000    10000       3    114     30     19

                    ##### concernant les éléments quadrangles #####
  115  2    10000    10000       4     44     34     13     52
  116  2    10000    10000       4    102     18     23     64
```

- *LMGC90* (type de fichier standard BODIES.DAT) :

```
                    ##### concernant les nœuds #####
NO2xx   1  coo1= 0.1800000D+00   coo2= 0.1750000D+00   coo3= 0.0000000D+00
NO2xx   2  coo1= 0.2650000D+00   coo2= 0.9000000D-01   coo3= 0.0000000D+00

                    ##### concernant les éléments triangles #####
          T3xxx      1 nodes     54     19     30
                     model  M2DNL  behav  bille
          T3xxx      2 nodes    114     30     19
                     model  M2DNL  behav  bille

                    ##### concernant les éléments quadrangles #####
          Q4xxx      3 nodes     44     34     13           52
                     model  M2DNL  behav  bille
          Q4xxx      4 nodes    102     18     23           64
                     model  M2DNL  behav  bille
```

L'utilisateur doit également veiller à ce que les données (conditions initiales et et modèle de calcul) de *LMGC90* et de *Code_Aster* soient identiques.

4.2 Préparation des commandes

Le couplage entre *Code_Aster* et *LMGC90* s'effectue via les macro-commandes MACR_LMGC_LINE et MACR_LMGC_NOLI. La première est utilisée dans le cas de comportements linéaires, la deuxième est utilisée dans le cas de comportements non-linéaires. Au niveau de la syntaxe, les deux macro-commandes sont identiques.

4.2.1 Syntaxe de la macro-commande

Couplage = MACR_LMGC_LINE (ou MACR_LMGC_NOLI)

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

- MODELE=MO, [modèle]
- CHAM_MATER=chmat [cham_mater]
- INCREMENT=_F([voir U4.51.03]
- CHAR_CINE=Char [char_cine]
- EXCIT=_F([voir U4.53.01]
  - CHARGE=char [char_meca]
  - FONC_MULT=fi [voir U4.51.03]

- THETA=coef_theta [voir U4.53.01]
- ETAT_INIT =_F ([voir U4.51.03]
  | VITE = vite, [cham_no_DEPL_R]
  | ACCE = acce, [cham_no_DEPL_R]
  ),
  =_F ([voir U4.51.03]
  |EVOL_NOLI = evol_noli, [evol_noli]
  |INST = time, [time reprise]
  |PRECISION = prec, [precision]
  ),
- BODIES=(_F(
  - GROUP_NO=group_no, [group_no]
  - LMGC90_INDEX=n, [number_mailx]
  - TYPE=
    /'LinearLagrangian', [corps_mailx]
    /'Lagrangian', [corps_mailx]
    /'NewtonEulerRigide2D' [corps_rig_2D]
    /'NewtonEulerRigide3D' [corps_rig_3D]
  ),
  _F(
  - GROUP_MA=group_ma, [group_ma]
  - LMGC90_INDEX=n, [number_mailx]
  - TYPE=
    /'LinearLagrangian', [corps_mailx]
    /'Lagrangian', [corps_mailx]
    /'NewtonEulerRigide2D' [corps_rig_2D]
    /'NewtonEulerRigide3D' [corps_rig_3D]
  ),
- ARCHIVAGE=_F([voir U4.51.03]
  - FREQ_DISPLAY=freq_dis, [freq_display]
  - FREQ_OUT_DOF=freq_out, [freq_out]
  ),

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

◆ INFO= /1 [DEFAUT]
/2 **);**

On ne détaille pas ici tous les mots-clés qui proviennent de *Code_Aster*. On se reportera aux commandes correspondantes [9, 11].

➢ BODIES： mot clé permettant à Saladyn de relier les corps maillés de *LMGC90* aux sous-maillages du modèle de *Code_Aster*. Le maillage de *Code_Aster* contient tous les corps maillés. Chaque mot-clé facteur ici décrit un corps maillé. Le corps maillé d'index LMGC90_INDEX dans *LMGC90* (donnée par BODIES.DAT) est le sous-maillage GROUP_NO (ou GROUP_MA) dans le modèle de *Code_Aster* , et réciproquement.

➢ TYPE：mot clé concernant le modèle mécanique de simulation. Les options de modèle sont "LinearLagrangian", ou "Lagrangian" correspondant respectivement à des modèles de calcul linéaires ou non-linéaires, corps par corps. .

➢ EXCIT：mot clé permettant de décrire les charges du modèle *Code_Aster* (sollicitations et conditions aux limites). Les opérandes ont la même signification que dans les documents [U4.51.03] et [U4.53.01].

➢ ARCHIVAGE=_F(: mot clé permettant d'archiver tous ou certains résultats à tous ou certains instants du calcul. En l'absence de ce mot clé tous les pas de temps sont archivés, y compris les instants de calculs nouvellement créés par redécoupage automatique du pas de temps (voir [U4.51.03] et [U4.53.01]). De plus, le mot clé est aussi utilisé pour archiver les résultats de *LMGC90*. L'option de FREQ_DISPLAY, qui contrôle l'archivage des résultats de la visualisation dans (/Results/DISPLAY/) tous les freq_dis pas de temps et l'option de FREQ_OUT_DOF qui contrôle l'archivage des fichier de bases dans (/Results/OUTBOX/) tous les freq_out pas de temps.
On développe un exemple ci-dessous afin de clarifier l'utilisation de ces différents mots-clés.

4.2.2 Exemple : contact de 3 billes déformables

Cet exemple traite de trois billes déformables (trois bodies) maillées par *Salomé* (fichier .med). Chaque bille est modélisée par 480 éléments triangles (cf. figure 2). Le comportement mécanique des billes déformables est simulé par *Code_Aster,* tandis que les interactions de contact frottement sont traitées par *LMGC90*.
Pour le couplage, on utilise la macro-commande suivante :

```
COUPLAGE=MACR_LMGC_NOLI(MODELE=MO,
                        CHAM_MATER=cha_mater,
                        INCREMENT=_F(LIST_INST=T_LIST),
                        CHAR_CINE=Char_cin,
                        EXCIT=(
                                    _F(CHARGE=asPesa,),
                                    _F(CHARGE=Load,),
                               ),
                        THETA= 0.61,
                        BODIES=(_F(GROUP_NO='Body_1',
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

```
                LMGC90_INDEX=1,
                                TYPE='LinearLagrangian',),
              _F(GROUP_NO='Body_2',
                LMGC90_INDEX=2,
                TYPE='LinearLagrangian',),
                                _F(GROUP_NO='Body_3',
                LMGC90_INDEX=3,
                TYPE='LinearLagrangian',),
              ),
ARCHIVAGE=(_F(FREQ_DISPLAY=100),
              _F(FREQ_DISPLAY=100),),
INFO=2,
)
```

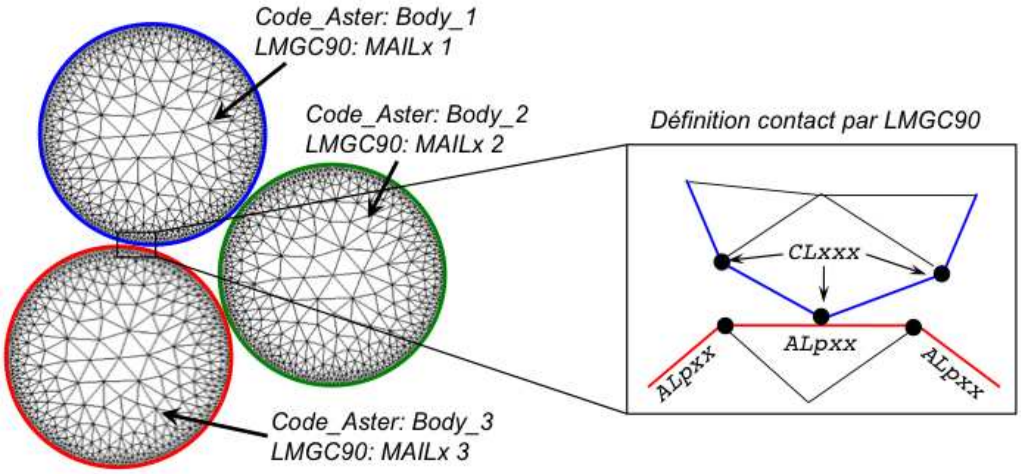


Figure 2: Géométrie du problème et la définition des forces de contact.

Par exemple, au niveau de la définition des forces d'interaction entre les billes ''bleu'' et ''rouge'' (cf. figure 2), *LMGC90* [4] va détecter le contact entre CLxxx et ALpxx via la commande suivante :

```
$seety
 cdbdy cdtac cdcol behav anbdy antac ancol      alert
 MAILx CLxxx TOURx gapc1 MAILx ALpxx UPxxx      0.1000000D-04
```

4.3 Lancement d'une étude avec couplage

Pour démarrer une étude avec couplage, on va lancer *Code_Aster* du module *SALADYN* sans l'interface graphique *ASTK*. Cette étape doit utiliser le fichier .export. Le contenu du fichier .export (par exemple yyyy.export) est le suivant :

```
F comm $pwd$/XXXX/yyyy.comm D  1
F msh  $pwd$/XXXX/yyyy.msh  D  19
R repe $pwd$/XXXX/Data      D  0
R repe $pwd$/XXXX/Results   R  0
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

```
F cmde $root_SALADYN$/Toolbox/TestsAster/Patch/aster.cmd D  0
F exec $root_SALADYN$/Toolbox/TestsAster/Patch/aster.exe D  0
R py   $root_SALADYN$/Toolbox/TestsAster/Patch/python    D  0
F mess $pwd$/XXXX/yyyy.mess R  6
F resu $pwd$/XXXX/yyyy.resu R  8
F resu $pwd$/XXXX/yyyy.med  R  80
F resu $pwd$/XXXX/yyyy.agr  R  37
F resu $pwd$/XXXX/yyyy.dat  R  39
```

Ensuite, à partir du répertoire XXXX, on va lancer *Code_Aster* du module *SALADYN* sans l'interface graphique par la commande as_run correspondant à $root_SALADYN$/aster/bin/as_run.

```
$pwd$/XXXX/
    ----> as_run yyyy.export
```

Après cette commande, le calcul va démarrer l'ensemble de *Code_Aster* et de *LMGC90*.

4.4 Résultats d'une étude avec couplage

Les résultats d'une étude avec couplage sont sauvegardés dans le répertoire /XXXX (résultats obtenus par *Code_Aster*).

```
$pwd$/XXXX/
    ----> .med; .msh; .agr; .dat; .mess; .resu; etc
```

Et ils sont sauvegardés dans le répertoire /XXXX/Results (résultats obtenus par *LMGC90*):

```
$pwd$/XXXX/Results/
    ----> /OUTBOX
    ----> /DISPLAY
    ----> /POSTPRO
```

5 Bibliographie

[1] F. Voldoire, Chaînage *LMGC90 – Code_Aster*,
Compte rendu EDF/R&D AMA [CR-AMA-06.195], 2006.

[2] F. Voldoire et M. Kham, Cas-test des structures basculantes et glissantes dans *Code_Aster*, Couplage *Code_Aster* & LMGC90, Compte rendu EDF/R&D AMA [CR-AMA-07.2894], 2008.

[3] Code_Aster :

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Site Internet: http://www.Code_Aster.org

[4] LMGC90 :
Site Internet: http://www.lmgc.univ-montp2.fr/~dubois/LMGC90/index.html

[5] M. Jean, The non smooth contact dynamics method, Compt.
Methods Appl. Math. Engrg, Vol. 177, pages 235–257, 1999.

[6] H-P. Cao, Modélisation par Éléments Discrets rigides et/ou déformables des milieux granulaires et des troisièmes corps solides. L'influence du comportement local sur le comportement global.
Thèse de l'INSA de Lyon, 2011.

[7] Siconos :
Site Internet : http://siconos.gforge.inria.fr/

[8] M. Abbas, Algorithme non linéaire quasi-statique (opérateur STAT_NON_LINE),
Manuel de référence de Code_Aster [R5.03.01], 2008.

[9] M. Abbas, Opérateur STAT_NON_LINE,
Manuel d'utilisation de *Code_Aster* [U4.51.03], 2010.

[10] N. Greffet, Algorithme non linéaire en dynamique,
Manuel de référence de *Code_Aster* [R5.05.05], 2010.

[11] N. Greffet, Opérateur DYNA_NON_LINE,
Manuel d'utilisation de Code_Aster [U4.53.01], 2010.

[12] A. Assire, Avancement de l'ANR *SALADYN* et construction d'un environnement développeur.
Compte rendu EDF/R&D/AMA [CR-AMA-11.303], 2011.

[13] Projet ANR Saladyn 2009-2012 : http://saladyn.gforge.inria.fr.

6 Description des versions du document

Date	**Version *Code_ Aster***	**Auteur(s) Organisme(s)**	**Description des modifications**
29/12/2011	11.2	H-P.CAO, F.VOLDOIRE LaMSID	Version initiale du document.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)