Module XMED - Guide de developpement

Version 0.1

G. Boulant

12 December 2011

Table des matières

(version pdf)

Ce document est la documentation technique du module XMED. Il fournit les instructions à suivre pour installer le module en vue d'un travail de développement, puis décrit les éléments de conception qui structurent le module.

Sommaire

Warning : Ce document est en travaux. Tant que cet avis n'aura pas disparu, veuillez en considérer le plan et le contenu encore incomplets, temporaires et sujets à caution.

CHAPITRE 1

Mise en place d'un espace de développement

1.1 Gestion de configuration du module XMED

Les sources du module (répertoire `xmed`) sont archivés en dépôt de configuration dans une base git du projet NEPAL. Ils peuvent être récupérés au moyen de la commande :

```
$ git clone git@cli70rw.der.edf.fr:xom/xmed.git
```

Cette commande installe un répertoire `xmed` contenant l'ensemble des sources du module XMED.

Le module XMED a pour pré-requis logiciel la plateforme SALOME :
– SALOME version 6.1.3 (au moins) à télécharger à l'URL http ://pal.der.edf.fr/pal/projets/pal/releases/V6_1_3
– On peut également utiliser une version dérivée comme SALOME-MECA 2010.1
– Installer la plate-forme choisie selon les instructions fournies.
Le module XMED utilise également une bibliothèque interne au projet NEPAL, appelée XSALOME, et qui fournit une extension aux fonctions de SALOME pour un usage de développement (XSALOME signifie eXtension SALOME). Les sources de cette bibliothèque doivent être récupérés au moyen de la commande :

```
$ git clone git@cli70rw.der.edf.fr:xom/xsalome.git
```

Cette commande installe un répertoire `xsalome` contenant l'ensemble des sources de la bibliothèque XSALOME.

Note : La bibliothèque XSALOME n'est pas un module SALOME mais une simple bibliothèque de fonctions qui complète ou rend plus facile d'utilisation les fonctions de SALOME. Elle NE DOIT EN AUCUN CAS être intégrée à d'autres projets que les projets internes NEPAL ou MAILLAGE. Il s'agit en effet d'une bibliothèque de transition qui héberge des développements destinés à être reversés dans la plate-forme SALOME. Le contenu et les interfaces de XSALOME ne peut donc être garanti sur le long terme.

1.2 Installation et lancement de l'application

L'installation suppose qu'une version 6.1.3 de SALOME (ou plus) est disponible et que le shell de travail est étendu avec l'environnement de SALOME. En général, par des commandes de la forme :

```
$ . /where/is/salome/prerequis.sh
$ . /where/is/salome/envSalome.sh
```

La compilation des modules xsalome et xmed suit le standard SALOME. La bibliothèque xsalome est un prérequis à la compilation de xmed. Pour cela, la variable d'environnement XSALOME_DIR doit être spécifiée pour la configuration de la procédure de reconstruction de xmed :

```
$ export XSALOME_DIR=<xsalome_installdir>
```

Aprés l'installation de xmed, il est possible de générer automatiquement une application SALOME prête à l'emploi pour la manipulation de champs :

```
$ <xmed_installdir>/bin/salome/xmed/appligen/appligen.sh
```

Cette commande génére un répertoire `appli` à l'emplacement où elle est exécutée. Il reste à lancer l'application SALOME au moyen de la commande :

```
$ ./appli/runAppli -k
```

1.3 Exécution des tests unitaires

Les tests unitaires peuvent être exécutés au moyen de scripts python lancés depuis une session shell SALOME. Dans un nouveau shell, taper :

```
$ ./appli/runSession
[NS=mars:2810]$ python appli/lib/python2.6/site-packages/salome/xmed/test_medoperation.py
```

L'exécution imprime un rapport détaillant le résultat pour chaque fonction de test :

```
test_addition (__main__.MyTestSuite) ... ok
test_arithmetics (__main__.MyTestSuite) ... ok
test_composition (__main__.MyTestSuite) ... FAIL
test_litteral_equation (__main__.MyTestSuite) ... ok
test_modification_of_attributes (__main__.MyTestSuite) ... ok
test_unary_operations (__main__.MyTestSuite) ... ok
test_update_metadata (__main__.MyTestSuite) ... ok
```

Les scripts de test sont :
- `test_medoperation.py` : tests des operations de champqs telles qu'elles sont mises en oeuvre depuis l'interface textuelle.
- `test_xmed.py` : tests des composants CORBA mis en oeuvre (`MEDDataManager` et `MEDCalculator`)

CHAPITRE 2

Conception générale

CHAPITRE 3

Eléments de conception détaillée

ANNEXE : Bug en cours

TO FIX :
– la composition d'opérations n'est pas possible (ex : 2*f1+f2) car 2*f1 est indiqué comme non compatible (il semble qu'il n'ai pas la reference correcte vers le maillage).
– le script de test test_medoperation.py plante si le module xmed n'a pas été chargé avec des données chargées.