Utilisation de CMake avec Morph-M

Matthieu Faessel

Séminaire outils Morph-M

-

19 juin 2008

Plan

1. Présentation de CMake
   - Généralités
   - Origines
   - Systèmes/Environnements supportés
   - Caractéristiques détaillées
   - Pourquoi utiliser CMake avec Morph-M ?
2. Utilisation
   - Cas général
   - Cas de Morph-M
3. Ecrire un projet CMake pour un module Morph-M
4. Documentation

1. Présentation de CMake
   - Généralités
   - Origines
   - Systèmes/Environnements supportés
   - Caractéristiques détaillées
   - Pourquoi utiliser CMake avec Morph-M ?
2. Utilisation
   - Cas général
   - Cas de Morph-M
3. Ecrire un projet CMake pour un module Morph-M
4. Documentation

Généralités

- Famille d'outils permettant de compiler, tester et archiver des sources/logiciels.
- Open-source et multi-plateformes.
- Développé par Kitware (2000).
- Supporte des environnements larges et complexes.

Origines

Bill Hoffman (2000), puis contributions de projets utilisant CMake (VXL, VTK, Paraview, . . .)

- Besoin d'un environnement de compilation multi-plateformes pour ITK
- Influence de **pcmaker** (VTK)
- Fonctionnalités de **configure** (Unix)

Utilisé aujourd'hui par de nombreux projets (KDE, Paraview, SecondLife, . . .).

Systèmes/Environnements supportés

Plateformes	Projets/Makefiles
Windows Mac OSX Linux i386 SunOS Sparc IRIX64 64 IRIX64 n32 HPUX 9000/785 AIX powerpc	Borland Makefiles MSYS Makefiles MinGW Makefiles NMake Makefiles Unix Makefiles Visual Studio 6 Visual Studio 7 Visual Studio 7 .NET 2003 Visual Studio 8 2005 (Win32 & Win64) Visual Studio 9 2008 (Win32 & Win64) Watcom WMake KDevelop3 XCode

Systèmes/Environnements supportés

À venir (version 2.6) :

- CodeBlocks - MinGW Makefiles
- CodeBlocks - Unix Makefiles
- Eclipse CDT4 - MinGW Makefiles
- Eclipse CDT4 - NMake Makefiles
- Eclipse CDT4 - Unix Makefiles

- Cross-compilation

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
  - tests de fonctionnalités du compilateur
  - recherche de chemins/fichiers *FIND_PATH*, *FIND_FILE*
  - recherche de librairies exécutables *FIND_LIBRARY*
  - recherche de packages *FIND_PACKAGE* (Boost, Jpeg, Png, Qt, VTK, ITK, Doxygen, Gnuplot, Latex, . . .)

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds

Caractéristiques détaillées

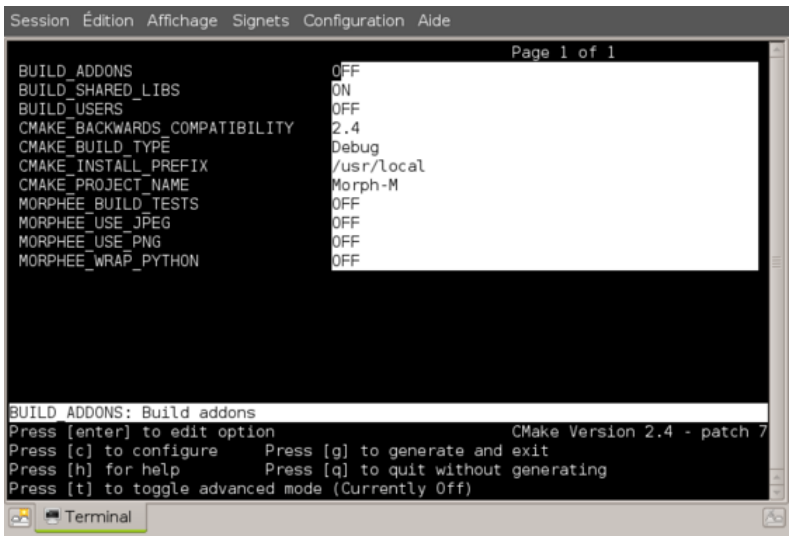
- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
  - plusieurs branches de compilation pour une même source
  - "propreté" du code

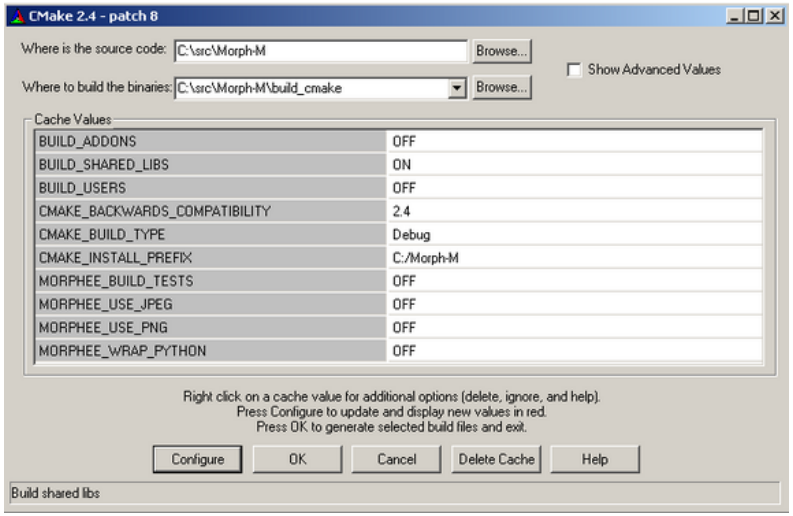
Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache





Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache
- Language de script simple

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache
- Language de script simple
- Permet de définir des cibles/règles personnalisées et complexes

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache
- Language de script simple
- Permet de définir des cibles/règles personnalisées et complexes
- Intègre un système de gestion des tests (CTest) et d'archives (CPack)

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache
- Language de script simple
- Permet de définir des cibles/règles personnalisées et complexes
- Intègre un système de gestion des tests (CTest) et d'archives (CPack)
- Intégration facile avec serveurs de test (CDash, Dart).

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache
- Language de script simple
- Permet de définir des cibles/règles personnalisées et complexes
- Intègre un système de gestion des tests (CTest) et d'archives (CPack)
- Intégration facile avec serveurs de test (CDash, Dart).
- Pas d'autre dépendance qu'un compilateur C++

Caractéristiques détaillées

- Intègre des outils permettant d'analyser la configuration du système
- In-place et out-of-place builds
- Interface + fichier de cache
- Language de script simple
- Permet de définir des cibles/règles personnalisées et complexes
- Intègre un système de gestion des tests (CTest) et d'archives (CPack)
- Intégration facile avec serveurs de test (CDash, Dart).
- Pas d'autre dépendance qu'un compilateur C++
- Liberal BSD licence

Pourquoi utiliser CMake avec Morph-M ?

- un seul projet pour toutes les plateformes
- possibilité de compiler son projet pour un système ou compilateur “exotiques”
- Macros spéciales Morph-M
  $\Rightarrow$ centralisation des paramètres de configuration
  $\Rightarrow$ simplification de l’écriture du projet

1 Présentation de CMake
- Généralités
- Origines
- Systèmes/Environnements supportés
- Caractéristiques détaillées
- Pourquoi utiliser CMake avec Morph-M ?

2 Utilisation
- Cas général
- Cas de Morph-M

3 Ecrire un projet CMake pour un module Morph-M

4 Documentation

Options I

BUILD_SHARED_LIBS Possibilité de compiler en SHARED (dll) ou en STATIC (lib).

CMAKE_BUILD_TYPE Configuration souhaité pour la compilation (Debug, Release,...).

MORPHEE_BUILD_TESTS Possibilité de compiler ou non les tests unitaires.
Nécessite la lib *boost_unit_test_framework*.

MORPHEE_WRAP_PYTHON Modules Pythons.
Nécessite les libs *python* et *boost_python*.

Options II

MORPHEE_USE_PNG et MORPHEE_USE_JPEG Prise en charge des librairies PNG et JPEG pour MorpheeIOExt.
Nécessite les libs *jpeg* et/ou *png*.

BUILD_ADDONS Compiler des modules addons.

BUILD_USERS Compiler des modules modules users.

1 Présentation de CMake
- Généralités
- Origines
- Systèmes/Environnements supportés
- Caractéristiques détaillées
- Pourquoi utiliser CMake avec Morph-M ?

2 Utilisation
- Cas général
- Cas de Morph-M

3 Ecrire un projet CMake pour un module Morph-M

4 Documentation

Ecrire un projet CMake pour un module Morph-M

Arborescence à respecter :

- un dossier *include* pour les headers (et *include/private* pour les templates),
- un dossier *src* pour les sources,
- un dossier *pythonExt* contenant les sources python,
- un dossier *test* contenant les sources des tests unitaires (les fichiers main.cpp, test_*.cpp, *_test.cpp ou *Tests.cpp).

Le fichier principal d'un projet CMake doit avoir comme nom CMakeLists.txt

Ecrire un projet CMake pour un module Morph-M

```
FIND_PACKAGE(Morphee REQUIRED)

SET(LIB_NAME ${MORPHEE_LIBS_PREFIX}operationResiduelles)
SET(LIB_DEPS
      ${MORPHEE_LIBS_PREFIX}common
      ${MORPHEE_LIBS_PREFIX}image
      ${MORPHEE_LIBS_PREFIX}imageIO
      ${MORPHEE_LIBS_PREFIX}imageIOExt
      ${MORPHEE_LIBS_PREFIX}stats
      ${MORPHEE_LIBS_PREFIX}selement
      ${MORPHEE_LIBS_PREFIX}morphoBase
      )

ADD_MORPHEE_LIBRARY(${LIB_NAME} ${LIB_DEPS})
ADD_MORPHEE_PYTHON_LIBRARY(${LIB_NAME} ${LIB_DEPS})
ADD_MORPHEE_UNIT_TEST(${LIB_NAME} ${LIB_DEPS})

IF(MSVC)
      IF(BUILD_SHARED_LIBS)
            SET_TARGET_PROPERTIES(${LIB_NAME} PROPERTIES COMPILE_FLAGS
                        -D__MORPHEE_DYNAMIC_QUASIDISTANCE_LIBRARY_EXPORT)
      ENDIF(BUILD_SHARED_LIBS)
ENDIF(MSVC)
```

Ecrire un projet CMake pour un module Morph-M

Inclure des sous-repertoires :

```
OPTION(MORPHEE_USE_ITK "Build ITK stufs")
OPTION(MORPHEE_USE_VTK "Build VTK stufs")

IF(MORPHEE_USE_ITK)
      ADD_SUBDIRECTORY(MorpheeITK)
ENDIF(MORPHEE_USE_ITK)

IF(MORPHEE_USE_VTK)
      ADD_SUBDIRECTORY(MorpheeVTK)
ENDIF(MORPHEE_USE_VTK)

ADD_SUBDIRECTORY(StochasticWatershed)
```

1 Présentation de CMake
- Généralités
- Origines
- Systèmes/Environnements supportés
- Caractéristiques détaillées
- Pourquoi utiliser CMake avec Morph-M ?

2 Utilisation
- Cas général
- Cas de Morph-M

3 Ecrire un projet CMake pour un module Morph-M

4 Documentation

Documentation

CMake :

`http://www.cmake.org/HTML/index.html`

CMake-Morph-M :

`http://cmm.ensmp.fr/~faessel/morphm_cmake/`