Tour d'horizon de CMake

Montel Laurent

Toulouse 26 janvier 2008

Qu'est ce qu'est « CMake » ?

C'est un logiciel qui est:

-> développé par Kitware (http://www.cmake.org)

-> Opensource

-> Multiplatforme (*Unix*/Mac/Windows)

-> un système de macros.

Utilité:

-> Un wrapper à la génération de Makefile.

-> Remplace automake/autoconf sous Unix.

-> génère les fichiers spécifiques aux environnements de compilation (Xcode/MSVC++/kdevelop3 etc.)

Les bases de CMake

Arborescence:

-> CMakeLists.txt

-> cmake/module/Find<name>.cmake

-> ConfigureCheck.cmake

-> config-<programme>.h.cmake

Les bases de CMake

Fonctionne sous forme de « Targets »

CMakeLists.txt

```
set(drkonqi_SRCS
   main.cpp
   debugger.cpp
   krashconf.cpp
   krashadaptor.cpp
   drbugreport.cpp
   backtrace.cpp
   toplevel.cpp )
kde4_add_executable(drkonqi ${drkonqi_SRCS})

target_link_libraries(drkonqi ${KDE4_KIO_LIBS})

install(TARGETS drkonqi DESTINATION ${LIBEXEC_INSTALL_DIR})
```

Les bases de CMake

Macro standards dans CmakeLists.txt:

-> add_subdirectory(<répertoire>)

-> find_package(<module)

-> option(<variable> <text>)

-> include (<fichier de macro>)

-> configure_file(config-<programme>.h.cmake <nom fichier généré>)

-> MESSAGE(STATUS <texte à afficher>)

Les bases de CMake

Module Find<name>.cmake

Variables standards:
-> <name>_FOUND
-> <name>_LIBRARIES
-> <name>_INCLUDE_DIRS

```
if (GMP_INCLUDE_DIR AND GMP_LIBRARIES)
  # Already in cache, be silent
  set(GMP_FIND_QUIETLY TRUE)
endif (GMP_INCLUDE_DIR AND GMP_LIBRARIES)
find_path(GMP_INCLUDE_DIR NAMES gmp.h )
find_library(GMP_LIBRARIES NAMES gmp )
if(GMP_INCLUDE_DIR AND GMP_LIBRARIES)
      set(GMP_FOUND 1)
endif(GMP_INCLUDE_DIR AND GMP_LIBRARIES)
mark_as_advanced(GMP_INCLUDE_DIR GMP_LIBRARIES)
```

Utilisation d'un module:
-> Ajouter: find_package(<nom du module>)
exemple: find_package(Eigen)

Les bases de CMake

ConfigureChecks.cmake

```
Image::Image(){
  qDebug()<<"Le système supporte les images de type :"
          <<"png,"
          <<"tiff";
}
```

Ajout dans CMakeLists.txt :

-> Find_package(PNG)

-> if(PNG_FOUND)

set(HAVE_PNG 1)

endif(PNG_FOUND)

-> configure_file(config-programme.h.cmake $

{CMAKE_CURRENT_BINARY_DIR}/config-programme.h)

Les bases de CMake

ConfigureChecks.cmake

Fichier config-programme.h.cmake:

```
#cmakedefine HAVE_PNG 1
```

il deviendra lors de la génération si PNG trouvé:

```
#define HAVE_PNG 1
```

sinon

```
// #define HAVE_PNG 1
```

```
#include <config-programme.h>

Image::Image(){
  qDebug()<<"Le système supporte les images:"
#ifdef HAVE_PNG
        <<"png"
#endif
        <<"tiff";
}
```

Les bases de CMake

ConfigureChecks.cmake

Recherche de fichier:

```
check_include_files(sys/stat.h    HAVE_SYS_STAT_H)
```

Recherche de prototype de fonction:

```
check_prototype_exists(mkstemps "stdlib.h;unistd.h" HAVE_MKSTEMPS_PROTO)
```

Compilation:

```
check_cxx_source_compiles("
  #include <sys/types.h>
  #include <sys/statvfs.h>
  int main(){
    struct statvfs *mntbufp;
    int flags;
    return getmntinfo(&mntbufp, flags);
  }
" GETMNTINFO_USES_STATVFS )
```

Les bases de CMake

Comment utiliser CMake ?

Configuration en ligne de commande:

lancer « cmake . » en ligne de commande :)

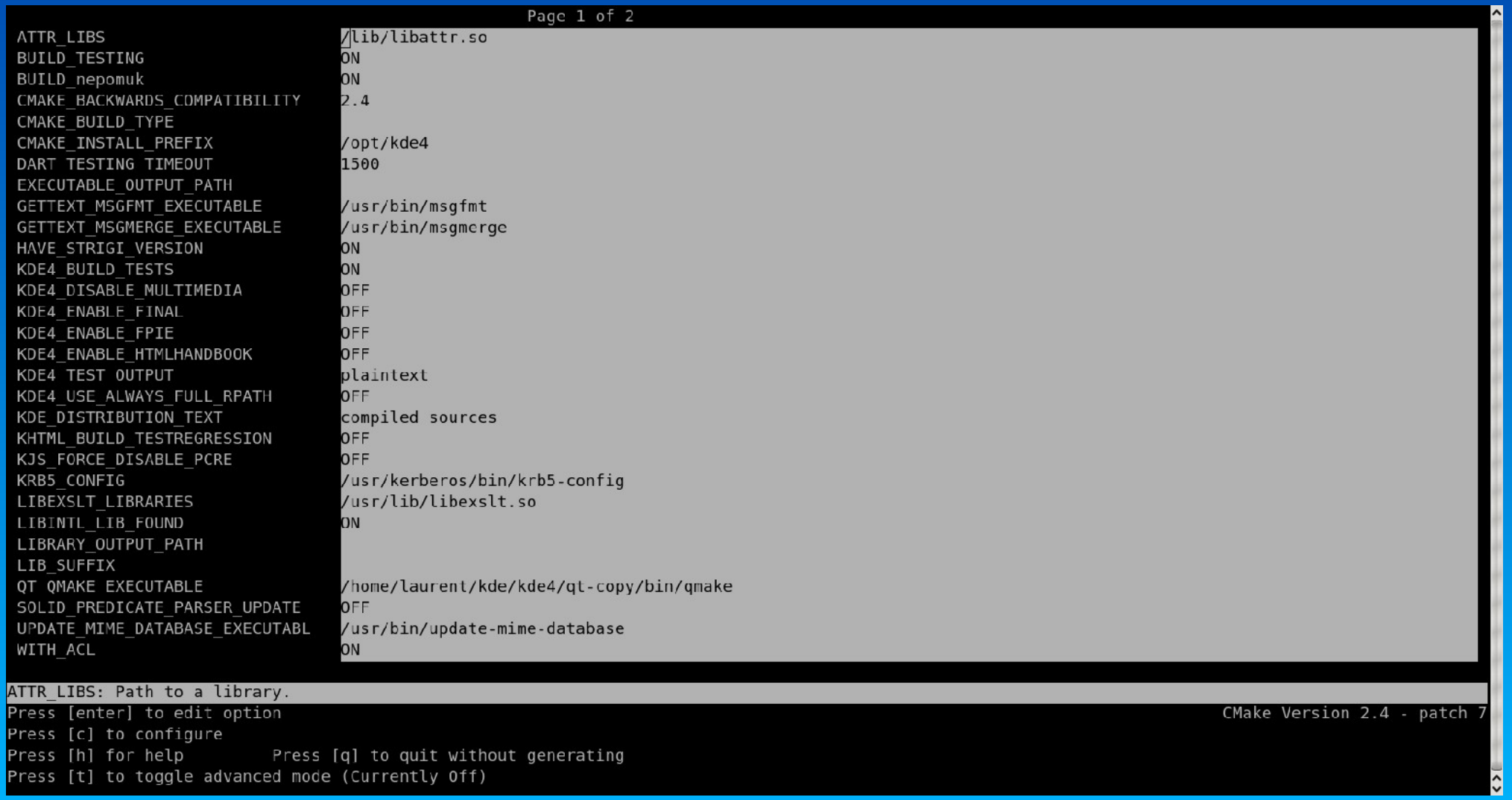
Passage d'options:

cmake -D<option> .

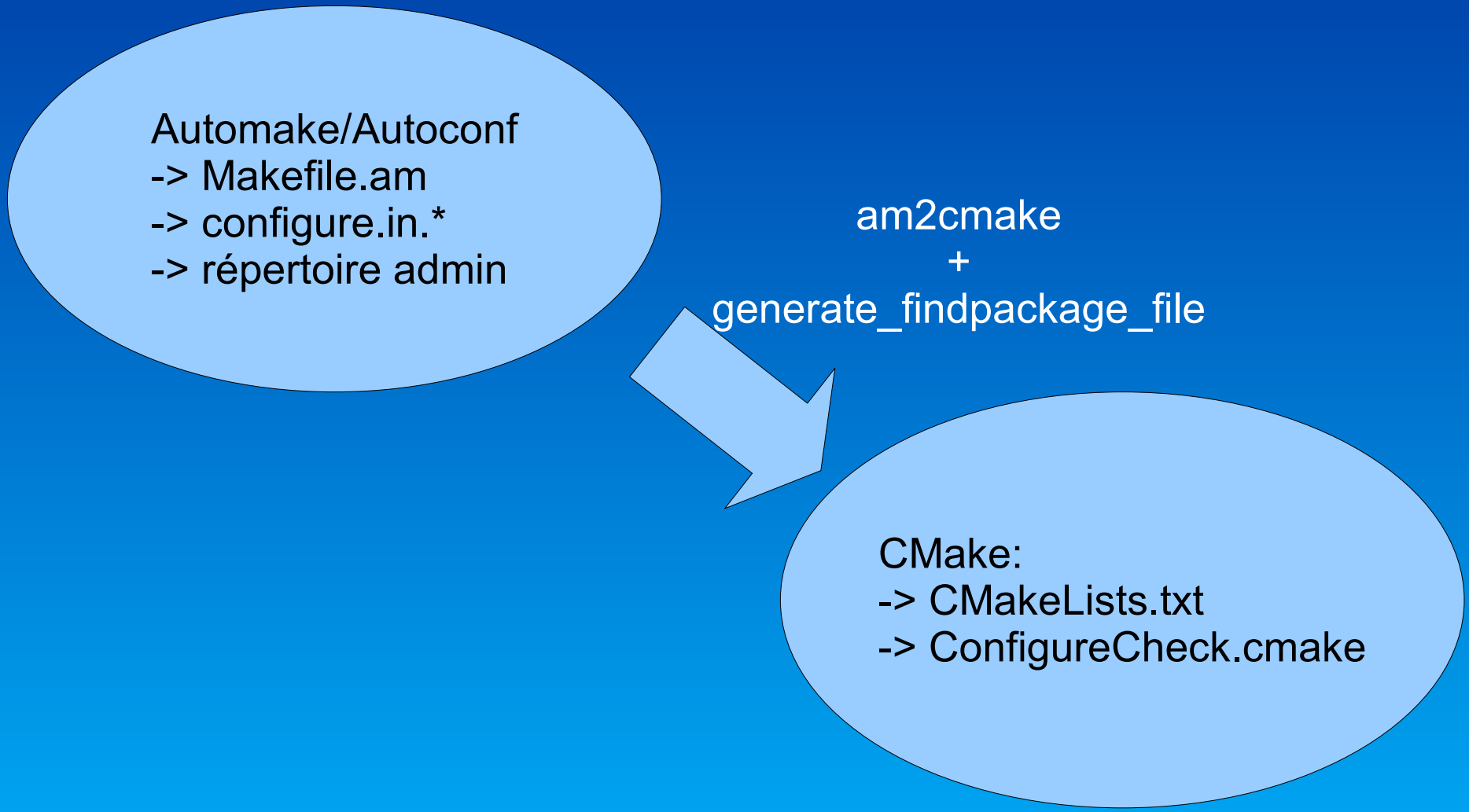
Exemple: cmake -DCMAKE_INSTALL_PREFIX=/opt/kde4 .

Les bases de CMake

CCMake ? (en ncurses)



CMake dans KDE



Questions ?