Working with CMT

Introduction

- Starting CMT
- Some examples
  - Starting from a trivial one
  - … to a real package
  - … while understanding (some of) the CMT concepts
- Discussing some issues (not all)
- Next questions to debate …

Working with CMT

Starting CMT

- Setup
  - `> . /afs/cern.ch/sw/contrib/CMT/v1r7/mgr/setup.sh`
  - `> source /afs/cern.ch/sw/contrib/CMT/v1r7/mgr/setup.csh`
- The main user interface
  - `> cmt [command <parameter ...>]`
  - `> cmt` (*without arguments*)
    - » Give the online help
- The most useful commands
  - `> cmt create <package> <version> [<location>]`
  - `> cmt show <option>`
    - » ***macro*(s), *set*(s), *constituent*(s), *uses*, *packages*, *versions*, …**
  - `> cmt broadcast [<selector>] <shell-command>`

Working with CMT

Examples…

- The simplest one
  - One new application with (some) source file(s)

1) Create a new package and move to it

2) Describe the package

3) Create the package items and build them

```
> cmt create A v1
> cd A/v1/cmt
> vi requirements

> vi A.cxx B.cxx C.cxx
> cmake
> . setup.sh
> ../${CMTCONFIG}/A.exe
```

```
package A
use AtlasRelease v2r1
application A A.cxx B.cxx C.cxx
```

4) set the environment and run

Working with CMT

A little break…

- The directory structure of a package

```
area / package-name / version-tag / cmt          / requirements
                                    src          / sources
                                    package-name / headers
                                    i386-linux22 / Linux binaries
                                    sun4u        / Sun binaries
                                    insure       / Insured binaries
                                    classes      / Java binaries
```

Public release
Private dev.
Common software
Inter-project com.
Domain specific dev.

Search list defined in CMTPATH

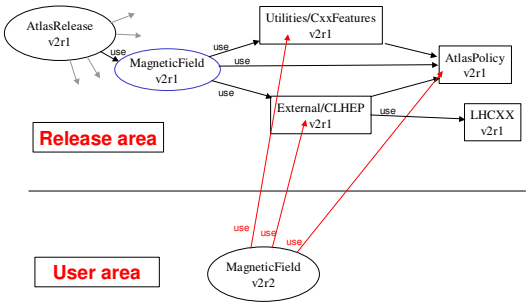
v1
v2r1
v3r1p1

Automatically created at build time and on demand according to `cmt system` or when providing a user defined tag (`gmake tag=insure`)
Available through the macro `$(<package-name>_tag)`

- Specified at package creation
- Obtained by `cmt checkout`

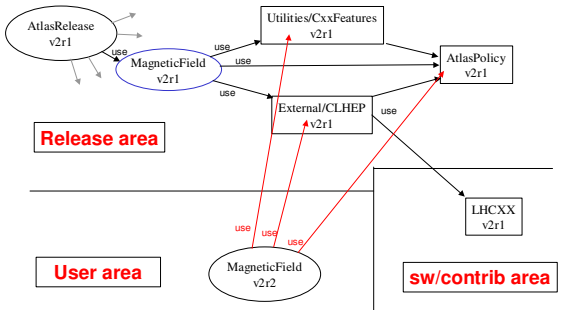
Working with CMT

Global software organization



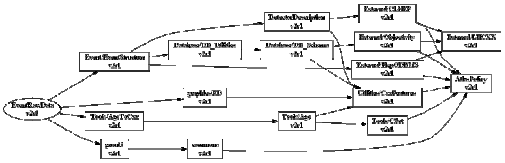
Working with CMT

Global software organization



Working with CMT

Showing the use graph



```
> cmt show uses -graph EventRawData v2r1 Event
```

This automatically reduces the apparent complexity of the use tree (by hiding the redundant use specifications)

Working with CMT

The MagneticField package

- This is the conversion of the original MagneticField package
- Obtained by:

```
> cd .../MagneticField
> cvs co -d v2r1 \
  -r offline-01-02-01 MagneticField
> cd v2r1
> mkdir cmt
> vi cmt/requirements
```

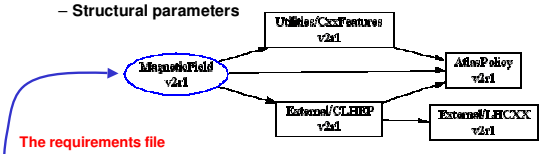
The requirements file

```
package MagneticField

author  Laurent Chevalier <laurent@hep.saclay.cea.fr>
author  Marc Virchaux <virchau@hep.saclay.cea.fr>

use AtlasPolicy v2r1
use CxxFeatures v2r1 Utilities
use CLHEP v2r1 External

include_dirs $(MAGNETICFIELDROOT)/MagneticField

branches MagneticField doc src test
...
```


Working with CMT

The MagneticField package

- Structural parameters



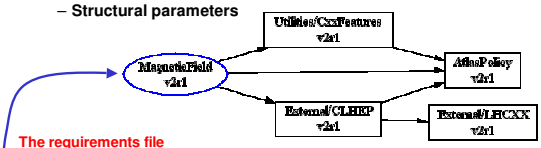
The requirements file

```
package MagneticField

author  Laurent Chevalier <laurent@hep.saclay.cea.fr>
author  Marc Virchaux <virchau@hep.saclay.cea.fr>

use AtlasPolicy v2r1
use CxxFeatures v2r1 Utilities
use CLHEP v2r1 External

include_dirs $(MAGNETICFIELDROOT)/MagneticField

branches MagneticField doc src test
...
```


Working with CMT

The MagneticField package

- Structural parameters



The requirements file

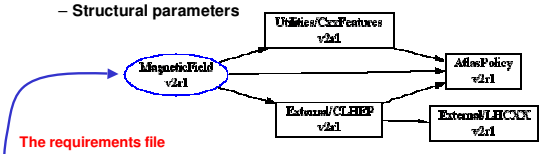
```
package MagneticField

author  Laurent Chevalier <laurent@hep.saclay.cea.fr>
author  Marc Virchaux <virchau@hep.saclay.cea.fr>

use AtlasPolicy v2r1
use CxxFeatures v2r1 Utilities
use CLHEP v2r1 External

include_dirs $(MAGNETICFIELDROOT)/MagneticField

branches MagneticField doc src test
...
```

Provides definitions for standard Atlas conventions (include paths, directory structure, default behavioural patterns, ...)

Working with CMT

The MagneticField package

- Structural parameters



The requirements file

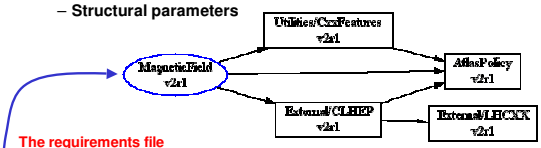
```
package MagneticField

author  Laurent Chevalier <laurent@hep.saclay.cea.fr>
author  Marc Virchaux <virchau@hep.saclay.cea.fr>

use AtlasPolicy v2r1
use CxxFeatures v2r1 Utilities
use CLHEP v2r1 External

include_dirs $(MAGNETICFIELDROOT)/MagneticField

branches MagneticField doc src test
...
```

An additional (non standard) include search path

Working with CMT

The MagneticField package

- Structural parameters



The requirements file

```
package MagneticField

author  Laurent Chevalier <laurent@hep.saclay.cea.fr>
author  Marc Virchaux <virchau@hep.saclay.cea.fr>

use AtlasPolicy v2r1
use CxxFeatures v2r1 Utilities
use CLHEP v2r1 External

include_dirs $(MAGNETICFIELDROOT)/MagneticField

branches MagneticField doc src test
...
```

Describes additional subdirectories (branches) specific to this package

Working with CMT

The MagneticField package

– Describing a constituent (a library)

```
...
apply_pattern default_no_share_linkopts

library MagneticField -no_share \
  AbstractMagneticField.cxx \
  MagField.cxx \
  MagFieldFor.cxx \
  MagFieldGradient.cxx \
  Tableau.cxx \
  reamag.F \
  thanatos.F

...
```

Apply a "pattern" (defined in ATlasPolicy):

Provide client packages with information needed to link with static library provided this package.

Working with CMT

The MagneticField package

– Describing a constituent (a library)

```
...
apply_pattern default_no_share_linkopts

library MagneticField -no_share \
  AbstractMagneticField.cxx \
  MagField.cxx \
  MagFieldFor.cxx \
  MagFieldGradient.cxx \
  Tableau.cxx \
  reamag.F \
  thanatos.F

...
```

This describes a (static) library and all its source files.

By default they are searched in ../src

The result will be

`libMagneticField.a`

Working with CMT

The MagneticField package

– Describing a test procedure

```
...
application test -check ../test/main.cxx

private

macro data_file "/afs/cern.ch/atlas/offline/data/bmagatlas02.data"

macro test_pre_check "ln -s $(data_file) test.dat"
macro test_check_args "test.dat"
macro test_post_check "/bin/rm -f test.dat"

macro test_dependencies MagneticField
```

Create an application named `test`, with one source file

run with the command

`> gmake check`

Working with CMT

The MagneticField package

– Describing a test procedure

```
...
application test -check ../test/main.cxx

private

macro data_file "/afs/cern.ch/atlas/offline/data/bmagatlas02.data"

macro test_pre_check "ln -s $(data_file) test.dat"
macro test_check_args "test.dat"
macro test_post_check "/bin/rm -f test.dat"

macro test_dependencies MagneticField
```

The following macro definitions are private to this package.

Client packages do not inherit these.

Working with CMT

The MagneticField package

– Describing a test procedure

```
...
application test -check ../test/main.cxx

private

macro data_file "/afs/cern.ch/atlas/offline/data/bmagatlas02.data"

macro test_pre_check "ln -s $(data_file) test.dat"
macro test_check_args "test.dat"
macro test_post_check "/bin/rm -f test.dat"

macro test_dependencies MagneticField
```

Define data file to be used in the test procedure.

Working with CMT

The MagneticField package

– Describing a test procedure

```
...
application test -check ../test/main.cxx

private

macro data_file "/afs/cern.ch/atlas/offline/data/bmagatlas02.data"

macro test_pre_check "ln -s $(data_file) test.dat"
macro test_check_args "test.dat"
macro test_post_check "/bin/rm -f test.dat"

macro test_dependencies MagneticField
```

These three standard make macros provide the parameters for the test procedure

Working with CMT

The MagneticField package

– **Describing a test procedure**

```
...
application test -check ../test/main.cxx

private

macro data_file "/afs/cern.ch/atlas/offline/data/bmagatlas02.data"

macro test_pre_check "ln -s $(data_file) test.dat"
macro test_check_args "test.dat"
macro test_post_check "/bin/rm -f test.dat"

macro test_dependencies MagneticField
```

Assure that `MagneticField` target is always built before the `test` target.

This is useful when using the `-j` option of `gmake`

Working with CMT

The main concepts

The requirements file centralizes *all* configuration parameters

Make macros and environment variables and their possible values on various platforms, sites, environments

Author(s), manager(s)

Customization for new languages, or document generators

The requirements file

Constituents
- Libraries
- Applications
- generated documents

Structural information
- specialized directory structure
- used packages
- links to external packages)

Definition of conventional behavioural patterns

Sufficient for transparently generating all required makefiles and setup scripts, or for launching test procedures. They are the sources of all queries.

Working with CMT

The main concepts

– **Package centric organization**

» **Every configuration item must be installed in the context of one package**

» **Configuration items are transmitted (~ *inherited*) along the use graph**

» **The concept of package is syntactically uniform...**

- **A CMT package is defined by its structure**

`<area>/<name>/<version-tag>/cmt/requirements`

» **But several semantic meanings can be assigned, eg:**

Normal packages	Contain sources for libraries, application, etc…
Policy packages	Specifications for project-wide policies
Interface packages	Offer a "CMT" touch to external packages
Release packages	The official and exhaustive set of packages validated to run together
Logical packages	Describe software domains (eg. Simulation, Reconstruction, …)

» **Allows project customization without changing CMT itself**

- **CMT provides generic and project-independent capabilities**

Working with CMT

Environment variables

- **Specified with the set or path keyword set**

`set set_append set_prepend set_remove`
`path_prepend path_append path_remove`

```
package LHCXX

set LHCXXTOP "" \
    CERN     "/afs/cern.ch/sw/lhcxx" \
    LAL      "/lal/lhcxx"

set LHCXX_PLATF "" \
    Linux "redhat61" \
    SunOS "sun" \
    HP-UX "hp" \
    OSF1  "dec"

set LHCXX_DIR "${LHCXXTOP}/specific/${LHCXX_PLATF}"
```

LHCXXTOP environment variable:

undefined except at CERN or at LAL.

Can be defined HERE for ALL Atlas sites.

Or, local versions of this interface package may be installed on remote sites.

Working with CMT

Environment variables

- **Specified with the set or path keyword set**

`set set_append set_prepend set_remove`
`path_prepend path_append path_remove`

```
package LHCXX

set LHCXXTOP "" \
    CERN     "/afs/cern.ch/sw/lhcxx" \
    LAL      "/lal/lhcxx"

set LHCXX_PLATF "" \
    Linux "redhat61" \
    SunOS "sun" \
    HP-UX "hp" \
    OSF1  "dec"

set LHCXX_DIR "${LHCXXTOP}/specific/${LHCXX_PLATF}"
```

Implement the conventions followed by the LHC++ software

Working with CMT

Environment variables

- **Specified with the set or path keyword set**

`set set_append set_prepend set_remove`
`path_prepend path_append path_remove`

```
package LHCXX

set LHCXXTOP "" \
    CERN     "/afs/cern.ch/sw/lhcxx" \
    LAL      "/lal/lhcxx"

set LHCXX_PLATF "" \
    Linux "redhat61" \
    SunOS "sun" \
    HP-UX "hp" \
    OSF1  "dec"

set LHCXX_DIR "${LHCXXTOP}/specific/${LHCXX_PLATF}"
```

This will provide base directory for LHC++ individual packages (eg CLHEP or HTL)

Working with CMT

Environment variables

- **Specified with the set or path keyword set**

  `set set_append set_prepend set_remove`
  `path_prepend path_append path_remove`

```
package CLHEP
use LHCXX v2r1 External

set CLHEP_VERSION "1.5.0.0"
set CLHEP_DIR "${LHCXX_DIR}/CLHEP/${CLHEP_VERSION}"

include_path "${CLHEP_DIR}/include"

macro CLHEP_linkopts "-L${CLHEP_DIR}/lib -lCLHEP"

path_append LD_LIBRARY_PATH "" \
                    Linux "${CLHEP_DIR}/lib"
```

This is an *interface* package to the physical CLHEP software

It should be site-independent thanks to the *installation* package

Working with CMT

Environment variables

- **Specified with the set or path keyword set**

  `set set_append set_prepend set_remove`
  `path_prepend path_append path_remove`

```
package CLHEP
use LHCXX v2r1 External

set CLHEP_VERSION "1.5.0.0"
set CLHEP_DIR "" CERN "${LHCXX_DIR}/CLHEP/${CLHEP_VERSION}"

include_path "${CLHEP_DIR}/include"

macro CLHEP_linkopts "-L${CLHEP_DIR}/lib -lCLHEP"

path_append LD_LIBRARY_PATH "" \
                    Linux "${CLHEP_DIR}/lib"
```

Set LD_LIBRARY_PATH only on Linux

Working with CMT

Make macros and symbols

- **Used to specify make macros, environment variables, aliases, path-like variables**
- **Alternate value can be specified for different sites, platforms, working conditions**

```
macro a "aaa" CERN "aax" LAL "aay"
macro b "bbb" Linux "bbx" alpha-osf40 "bby"
macro c "ccc" debug "ccx" insure "ccy"
```

```
macro_append
macro_prepend
macro_remove
```

- **They can be edited**
- **They are automatically valued when gmake runs**
- **They can be evaluated**

```
> cmt show macro cppflags
> cmt show macro_value cppflags -tag=alpha
```

- **CMT provides a wide set of macros for standard parameterization (eg cppflags, fflags, linkopts, etc…)**

Working with CMT

The evaluation

- **Originally done onto offline-01-01-05 (june-july 2000)**
- **The work has been done on a clone of the CVS repository**
- **All requirements files for packages in offline-01-01-05 were ready (fully functional)**
- **Two parallel build were done (one with CMT and one with SRT)**
- **No visible harm was detected onto that release**

- **Now the work is being replicated for offline-01-02-01 (ie. tuning previous requirements files and filling in those for newly introduced packages)**
- **They could then be introduced *smoothly* in the standard CVS repository**
- **Users would then simply have to *use* or to *upgrade* them**

Working with CMT

How to work with the release

- **Here, we consider the situation of a functional release managed by CMT**

```
export CMTPATH=/afs/cern.ch/atlas/release
export CMTSITE=CERN
```

1. **Package modification**

```
> cd mydevarea
> cmt checkout MagneticField
> cd MagneticField/v2r1/cmt
> . setup.sh
> gmake
```

2. **Package creation**

```
> cd mydevarea
> cmt create A v1
> cd A/v1/cmt
> vi ...
> . setup.sh
> gmake
> cd ..
> cvs import -m "…" A Atlas v1
```

```
package A

use MagneticField v2r1

application myapp *.cxx
```


Working with CMT

Next points to discuss…

- **The role and meaning of the version tag**
  - **The question of backward compatibility**
  - **Incremental releases**
- **The role of software areas**
  - **Connection to external packages (Gaudi)**
  - **Development areas**
- **Software export**
  - **Through remote CVS (cmt checkout)**
  - **Through AFS (and CMTPATH)**
  - **Remote site configuration through CMTSITE**
- **Defining Atlas specific patterns**
  - **Database management**
  - **Test procedures**