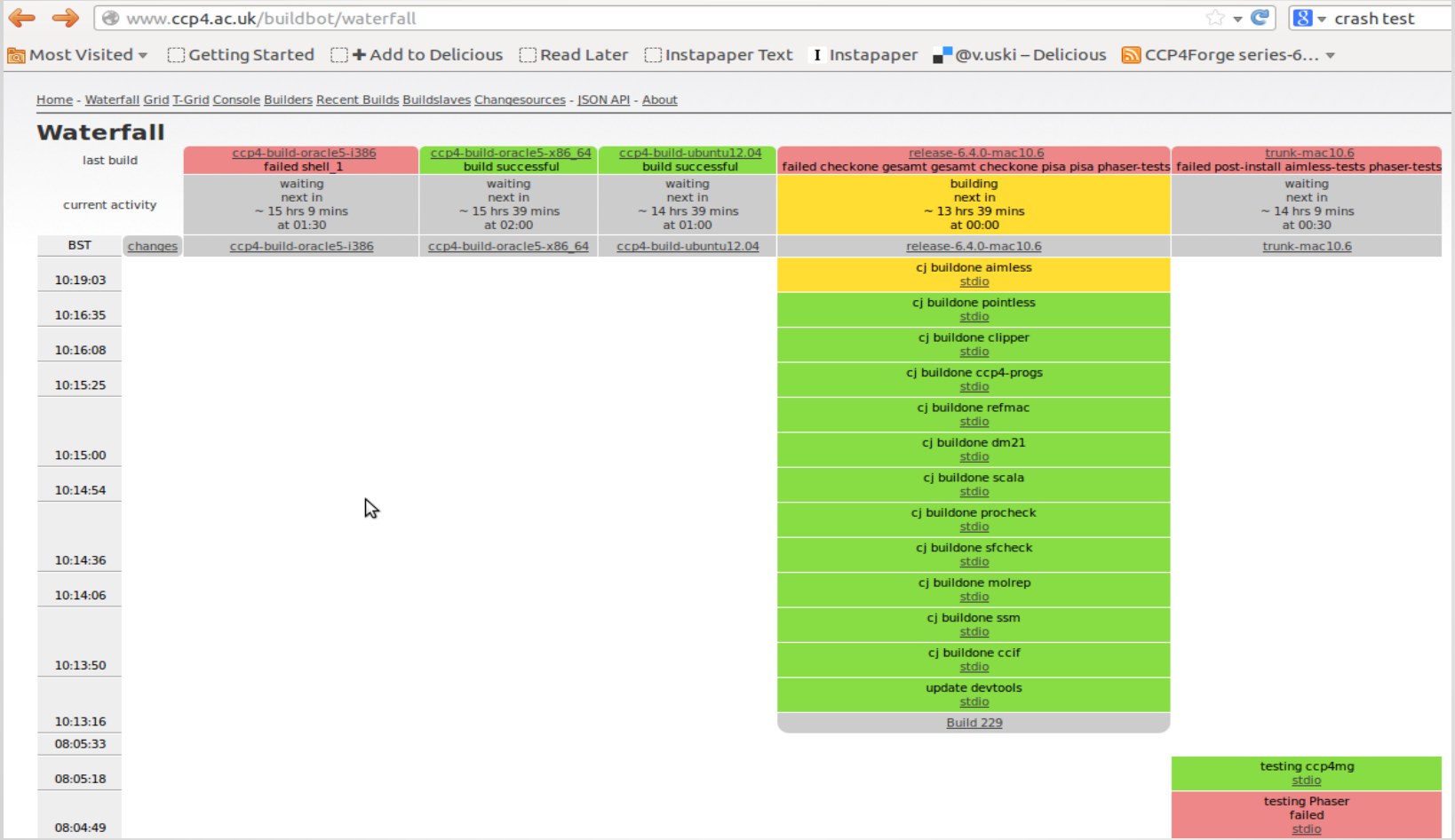
Build/Update/Tests

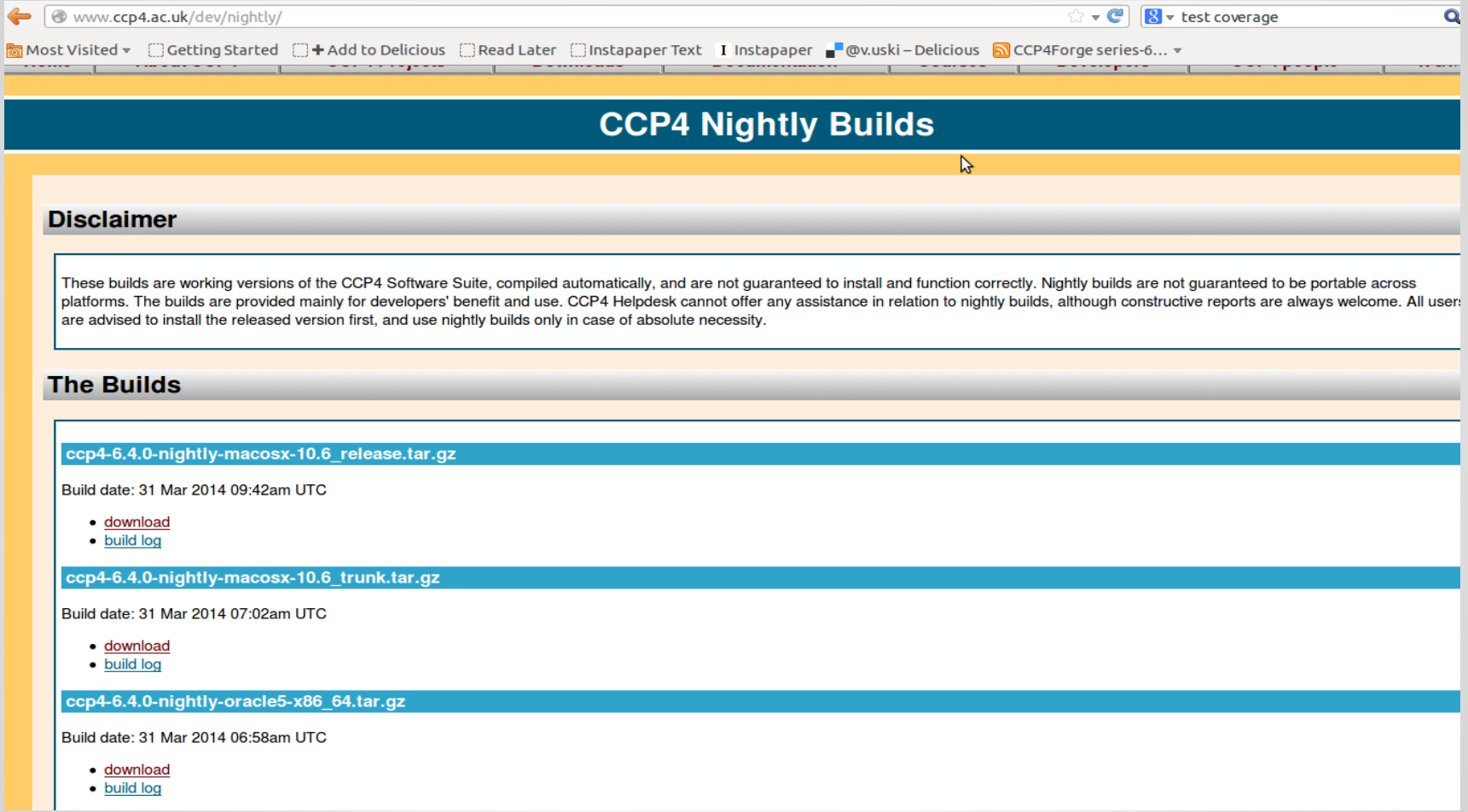
Nightly builds for CCP4

- automatic builds on all supported platforms
- use the new build system (jhbuild)
- powered by Buildbot

Goals:

- find build errors quickly
- find regressions quickly
- make testing easier


www.ccp4.ac.uk/buildbot/waterfall
crash test
Most Visited
Getting Started
Add to Delicious
Read Later
Instapaper Text
Instapaper
@v.uski – Delicious
CCP4Forge series-6...
Home - Waterfall Grid T-Grid Console Builders Recent Builds Buildslaves Changesources - JSON API - About
Waterfall
last build
ccp4-build-oracle5-i386 failed shell_1
ccp4-build-oracle5-x86_64 build successful
ccp4-build-ubuntu12.04 build successful
release-6.4.0-mac10.6 failed checkone gesamt gesamt checkone pisa pisa phaser-tests
trunk-mac10.6 failed post-install aimless-tests phaser-tests
current activity
waiting next in ~ 15 hrs 9 mins at 01:30
waiting next in ~ 15 hrs 39 mins at 02:00
waiting next in ~ 14 hrs 39 mins at 01:00
building next in ~ 13 hrs 39 mins at 00:00
waiting next in ~ 14 hrs 9 mins at 00:30
BST
changes
ccp4-build-oracle5-i386
ccp4-build-oracle5-x86_64
ccp4-build-ubuntu12.04
release-6.4.0-mac10.6
trunk-mac10.6
10:19:03
cj buildone aimless
stdio
10:16:35
cj buildone pointless
stdio
10:16:08
cj buildone clipper
stdio
10:15:25
cj buildone ccp4-progs
stdio
cj buildone refmac
stdio
10:15:00
cj buildone dm21
stdio
10:14:54
cj buildone scala
stdio
cj buildone procheck
stdio
10:14:36
cj buildone sfcheck
stdio
10:14:06
cj buildone molrep
stdio
cj buildone ssm
stdio
10:13:50
cj buildone ccif
stdio
update devtools
stdio
10:13:16
Build 229
08:05:33
08:05:18
testing ccp4mg
stdio
08:04:49
testing Phaser failed
stdio


www.ccp4.ac.uk/dev/nightly/
test coverage
Most Visited
Getting Started
Add to Delicious
Read Later
Instapaper Text
Instapaper
@v.uski – Delicious
CCP4Forge series-6...
CCP4 Nightly Builds
Disclaimer
These builds are working versions of the CCP4 Software Suite, compiled automatically, and are not guaranteed to install and function correctly. Nightly builds are not guaranteed to be portable across platforms. The builds are provided mainly for developers' benefit and use. CCP4 Helpdesk cannot offer any assistance in relation to nightly builds, although constructive reports are always welcome. All users are advised to install the released version first, and use nightly builds only in case of absolute necessity.
The Builds
ccp4-6.4.0-nightly-macosx-10.6_release.tar.gz
Build date: 31 Mar 2014 09:42am UTC
download
build log
ccp4-6.4.0-nightly-macosx-10.6_trunk.tar.gz
Build date: 31 Mar 2014 07:02am UTC
download
build log
ccp4-6.4.0-nightly-oracle5-x86_64.tar.gz
Build date: 31 Mar 2014 06:58am UTC
download
build log

Nightly builds - current status

developers branches built on

- Oracle Linux 5.9, 32 and 64 bit, Ubuntu
- Mac OS X 10.6

release bundles on

- Mac OS X 10.6
- CentOS 5 Linux (Marcin’s builds)
- Windows (“)

Nightly builds - coming soon

Computer resources provided by SESP (Software Engineering Support Programme) at RAL

1. many compilers/compiler versions
2. various platforms (Linux/Windows)
3. Goal: Set up a true “build brigade” for CCP4

building your own

1. Install devtools (to build the dev branches):

```
bzr checkout \ http://fg.oisin.rc-harwell.ac.
uk/anonscm/bzr/devtools/ devtools
```

or (to build the release bundle)

```
bzr checkout \ http://fg.oisin.rc-harwell.ac.
uk/anonscm/bzr/series-64/devtools
    devtools
```

building your own..

2. edit cj.rc if required

```
os.environ['PYTHON'] = "/path/to/bin/python"
os.environ['CC'] = "icc"
os.environ['CXX'] = "icpc"
os.environ['FC'] = "ifort"
os.environ['QT_QMAKE_EXECUTABLE'] = "/usr/local/Trolltech/Qt-4.8.4/bin/qmake"
os.environ['LDSHARED'] = "icc -Wl,-F. -bundle -undefined dynamic_lookup"
os.environ['MACOSX_DEPLOYMENT_TARGET'] = "10.6"
jobs = 4
module_cmakeargs['setup'] = cmakeargs + ( " -DQT_QMAKE_EXECUTABLE=/usr/local/Trolltech/Qt-4.8.4
/bin/qmake")
skip += [ "lapack", "tcltk84", "pyqt4", "libxml2", "zlib", "libxslt"]
```

building your own..

3. build everything (e.g. for Mac):

```
$ ./cj build ccp4-osx
```

or build single programs:

```
$ ./cj build refmac
```

For more info, see: `devtools.fg.oisin.rc-harwell.ac.uk`

Test suite

- Tests for the following programs are run with the nightly builds: CRANK, Refmac, Aimless, Pointless, Phaser, CCP4mg
  - most common options
  - integration testing (useful for pipelines)
- run-all script (smoke tests for most programs)
- Wanted: More tests and more testing!

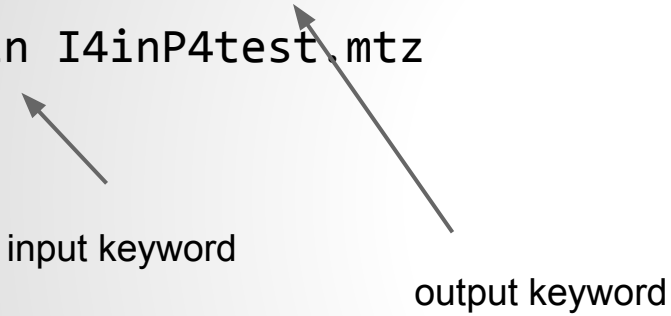
Test system

- framework for automated testing of CCP4 programs
- developed by Pavol Skubak
- To write a new test:

1. Write a script to run the test.
2. Define keywords (input, output, “extra”).
3. Identify the testing criteria.

example script file

```
pointless xmlout latcen.xml <<EOF
hklin I4inP4test.mtz
EOF
```



example test parameters (crit.def)

```
_crash_   desc      "Has the job crashed or not"
_crash_   getdef    crash_status
_crash_   optimum   0
_crash_   diff      0

bestsolutiongroup getxml out["xmlout"] POINTLESS/BestSolution/GroupName
bestsolutiongroup descr Space group or Laue group determined

bestsolutionTotalProb getxml out["xmlout"] POINTLESS/BestSolution/TotalProb
bestsolutionTotalProb descr  Total probability of group solution
```

example test criteria (crit.def)

```
bestsolutiongroup          optim      “P 42 21 2”
bestsolutiongroup          diff       0

bestsolutionTotalProb      optim      0.823
bestsolutionTotalProb      diff       0.05
bestsoltionTotalProb       regdiff    0.01
```

importing and running the tests

$ testsys.py import -p pointless

$ testsys.py run -p pointless

Information from run of test various , prog pointless: Test prepared for run in directory /home/ville/Test-Framework/testsys-tests/runs/pointless/various/155

Going to submit jobs from test lauegroup

Submitted job latcen (pid 8484).

Information from evaluation of job latcen , test lauegroup , prog pointless: Determination of output variables started.

EVALUATION OF RUN 155

All OK!

installing the test system

bzr checkout \ http://fg.oisin.rc-harwell.ac.uk/anonscm/bzr/test-sys/trunk test-sys

(includes full documentation)

Submitting new tests

bzr repo:

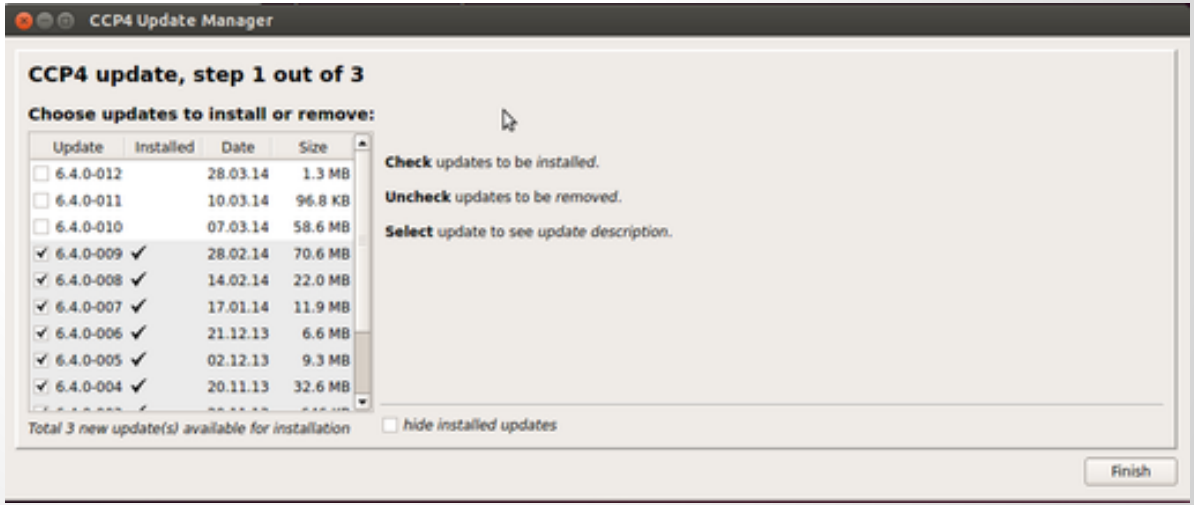
`bzr+ssh://username@fg.oisin.rc-harwell.ac.uk/var/lib/gforge/chroot/scmrepos/bzr/tests/trunk`

‘bzr add’ the new files in the (import and) xml dirs (the latter are created after importing) and email the data to ville.uski@stfc.ac.uk (ask for more details)

Updates

Testing before update release: Use the expert mode

ccp4um -expert



Working with Bzr (distributed style)

1. $ bzr branch \ bzr+ssh://username@fg.oisin.rc-harwell.ac.uk/var/lib/gforge/chroot/scmrepos/bzr/mmdb/trunk mmdb
2. $ cd mmdb
3. edit files
4. $ ./configure && make
5. test it
6. $ bzr commit -m “description of the new revision”
7. repeat steps 3-6 if desired
8. $ bzr pull :parent # pull revisions committed by others
9. $ bzr push :parent # if there are no conflicts to resolve

Working with Bzr (centralised style)

1. `$ bzr checkout \ bzr+ssh://username@fg.oisin.rc-harwell.ac.uk/var/lib/gforge/chroot/scmrepos/bzr/mmdb/trunk mmdb`
2. `$ cd mmdb`
3. `edit files`
4. `$ ./configure && make`
5. `test it`
6. `$ bzr update`
7. `$ bzr commit -m “description of the new revision”`

This will automatically update the parent branch.

If the branches have diverged

1. `$ bzr pull :parent`

   ```
   bzr: ERROR: These branches have diverged.
   ```

2. `$ bzr merge :parent`

   ```
   M  README
   Text conflict in README 1 conflicts encountered.
   ```

3. `$ vi README`

   ```
   <<<<<<< TREE
   the original text
   =======
   the new text
   >>>>>>> MERGE-SOURCE
   ```

resolving the conflict

1. `$ vi README`

   the original and new text merged

2. `$ bzr resolve README`
3. `$ bzr commit -m “description”`
4. `$ bzr push :parent`

other common tasks

adding a new file:

```
bzr add FILE && bzr commit
```

removing a file:

```
bzr rm FILE && bzr commit
```

renaming a file:

```
bzr mv FILE FILE2 && bzr commit
```