Buildbot :

Le logiciel utilisé pour compiler et tester automatiquement les développements réalisés dans le coupleur OASIS3-MCT

L. Coquart, I. D'Ast, S. Valcke

UMR CECI 5318, CNRS-CERFACS, TR-CMGC-17-85 May 2017

Table des Matières

1 Introduction

BuildBot est un système d'automatisation des cycles de compilation et des tests nécessaires à la plupart des projets logiciels pour valider les modifications de code. En recompilant et testant automatiquement les sources à chaque modification, les problèmes sont rapidement découverts et peuvent être corrigés, avant que d'autres développeurs ne soient confrontés à l'erreur. Le logiciel a de plus une interface graphique qui permet de visualiser très rapidement et très simplement les résultats des tests : vert si tout s'est bien passé, rouge s'il y a eu un problème.

Buildbot est un système réparti, suivant le modèle ***Buildmaster*** (maître)/***Buildslave*** (esclave). Toute la configuration et les définitions sont faites au niveau du ***Buildmaster*** (maître), les ***Buildslaves*** (esclaves) ne faisant qu'exécuter les commandes transmises par le maître et lui renvoyer les résultats comme le montre la Figure 1 ci-dessous. ***Buildbot*** est écrit en python et utilise le protocole twisted (http et ssh).

Buildbot est utilisé au CERFACS-CNRS dans le cadre du développement du coupleur OASIS3-MCT (Valcke et al., 2015). Ce coupleur est très répandu dans la communauté climatique européenne (environ dans trente groupes) et dans la communauté internationale, où il est utilisé pour coupler différentes composantes du système Terre. Il est nécessaire de le tester avant de rendre disponible une nouvelle version officielle aux utilisateurs. Le code d'OASIS3-MCT est géré sous le gestionnaire de révisions SVN (Subversion). ***Buildbot*** permet de recompiler automatiquement les sources d'OASIS3-MCT lorsqu'un développement est réalisé sur le tronc ou une branche de SVN, puis de réaliser un ensemble de nombreux tests avec des modèles jouets pour détecter si la modification n'a pas introduit de bug. Ces modèles jouets ne contiennent aucune physique mais reproduisent les algorithmes de couplage nécessaires pour tester les différentes fonctionnalités du coupleur ou reproduisent les algorithmes des vrais modèles couplés des utilisateurs.

La partie 2 présente le fonctionnement général de ***Buildbot***. Les concepts de base et le vocabulaire utilisés dans le fichier de configuration ***master.cfg*** et dans l'interface graphique sont présentés dans la partie 2.1. Puis la partie 2.2 décrit comment créer le ***Buidmaster*** (maître) et les ***Buildslaves*** (esclaves) et la partie 2.3 comment les lancer puis les arrêter. La partie 2.4 présente l'interface graphique de ***Buildbot*** qui est une composante fondamentale du logiciel pour analyser les résultats des tests. Elle permet de lancer et d'arrêter les tests « à la main » ou de lancer des tests « automatiquement » après une mise à jour du code sous SVN, puis de visualiser très rapidement les résultats des tests. La partie 3 s'intéresse aux résultats obtenus pour le tronc d'OASIS3-MCT. La partie 3.1 définit précisément les tests réalisés et la partie 3.2 les résultats obtenus dans ce cas particulier.

2 Présentation générale de Buildbot

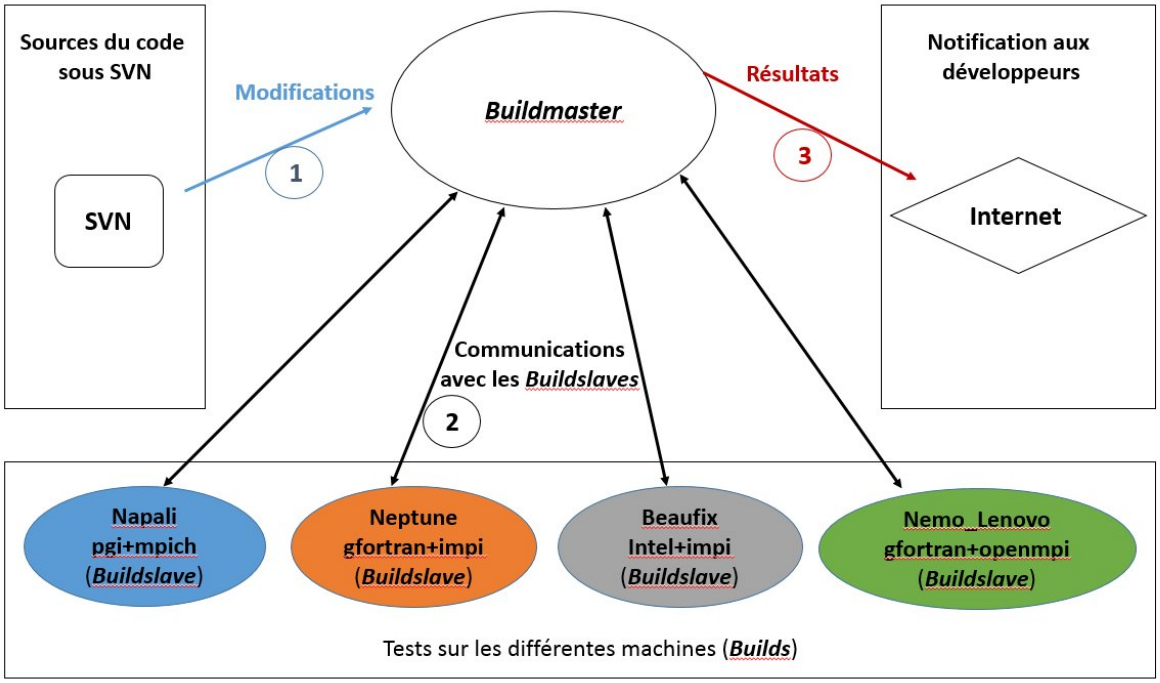


Figure 1 : Fonctionnement de Buildbot avec OASIS3-MCT

2.1 Concepts de base et vocabulaire

Le ***buildmaster*** (maître) lit le fichier de configuration ***master.cfg***, code python définissant un dictionnaire ***BuildmasterConfig***, via des clés. Comme le montre la Figure 1, il « écoute » le serveur SVN. Quand il y a une mise à jour dans les sources sous SVN, il transfère ces infos aux ***Schedulers*** (planificateurs) qui gèrent le lancement des ***Builders*** (tests) sur chaque ***Buildslave*** (machine de travail esclave). Les ***Builders*** correspondent à la description exacte des tests à faire (***steps***). L'ensemble des tests à faire sur un ***Buildslave*** s'appelle un ***Build*** (c'est donc l'ensemble des ***Builders*** sur un ***Buildslave***). Ces termes seront utilisés dans le fichier de configuration ***master.cfg*** et dans l'interface graphique.

En général les ***Builders*** (tests) se font après la mise à jour du code sous SVN, mais il est toujours possible de déclencher les ***Builders*** (tests) et de les arrêter via l'interface graphique, comme on le verra dans la partie 2.4.

Le fichier de configuration, ***master.cfg***, utilisé actuellement pour tester le tronc d'OASIS3-MCT, qui sera la prochaine version officielle OASIS3-MCT_4.0 quand tous les développements planifiés seront terminés et testés sous ***Buildbot***, est détaillé en Annexe A.

Il est nécessaire de séparer les résultats des tests en fonction des versions du coupleur car le fichier de configuration ***master.cfg*** évolue avec la version d'OASIS3-MCT. En effet, les sources ne sont pas stockées au même endroit sous SVN (chaque version stable du coupleur est stockée sur sa branche SVN où ne sont corrigés que les bugs importants), les ***Builders*** (tests) à effectuer ne sont pas les mêmes, et les ***Buildslaves*** (machines esclaves) ne sont pas les mêmes.

Par exemple, il y a plus de fonctionnalités à tester pour OASIS3-MCT_3.0 (Valcke et al., 2015) que pour OASIS3-MCT_2.0 (Valcke et al., 2013) ou OASIS3-MCT_1.0 (Valcke et al., 2012), et certains des ***Buildslaves*** (machines esclaves) utilisés pour OASIS3-MCT_1.0 n'existent plus (HP proliant Corail au CERFACS, NEC à Météo-France).

Pour chaque nouvelle version officielle du coupleur, OASIS3-MCT_1.0, OASIS3-MCT_2.0 et OASIS3_MCT_3.0, les résultats des tests sous ***Buildbot*** de chaque version ont donc été regroupés sous un même répertoire oasis3-mct_1.0, puis oasis3-mct_2.0 et enfin oasis-mct_3.0. Dans chacun de ces répertoires, un sous-répertoire maître correspondant à la version a été créé, master_oa3-mct_1.0 pour 1.0, master_oa3-mct_2.0 pour 2.0 et master_oa3-mct_3.0 pour 3.0, ainsi que les sous-répertoires correspondants aux ***Buildslaves*** (machines esclaves associées) sur lesquels les tests ont été effectués.

Dans la suite du rapport, nous nous intéressons aux résultats obtenus sur quatre ***Buildslaves*** (machines esclaves) différents, obtenus avec les sources du tronc d'OASIS3-MCT, qui est la version en développement et qui correspond à la future version officielle OASIS3-MCT_4.0. Dès que cette version inclura tous les développements prévus, sera stable et aura été testée avec ***Buildbot***, elle sera stockée sur une nouvelle branche, où seuls les bugs importants seront corrigés. Cette version sera accessible aux utilisateurs à l'adresse :

https://oasis3mct.cerfacs.fr/svn/branches/OASIS3-MCT_4.0_branch/oasis3-mct/

2.2 Création du Buildmaster (maître) et des Buildslaves (esclaves)

Pour créer l'environnement sous ***Buildbot*** qui sert à tester le tronc d'OASIS3-MCT, on commence par créer le répertoire oasis3-mct_tronc :

- mkdir oasis3-mct_tronc ; cd oasis3-mct_tronc

Puis on crée le ***Buildmaster*** (maître) :

- ***buildbot create-master*** master_oa3-mct_tronc

Le fichier ***master.cfg.sample***, qu'il faut adapter à nos tests et recopier dans ***master.cfg***, est alors créé dans le répertoire master_oa3-mct_tronc.

Puis on crée les quatre esclaves qui correspondent aux ***Buildslaves*** (machines esclaves) sur lesquels on fait les tests, comme le montre la Figure 1. **Napali** est un serveur Linux

situé au CERFACS, avec 2 procs Xeon chacun de 6 cœurs et avec une mémoire totale de 198 Go. Le compilateur utilisé pour faire les tests est pgi avec la librairie MPI mpich. **Neptune** est un cluster BULL situé au CERFACS. Il est constitué de 158 nœuds de calcul bi-socket SandyBridge chacun avec 16 cœurs dotés de 32 Go de mémoire. Le compilateur utilisé pour faire les tests est gfortran avec la librairie Intel MPI impi. **Nemo_Lenovo** est un serveur Lenovo situé au CERFACS. Il est constitué de 288 nœuds de calcul bi-socket Intel Haswell chacun avec 24 cœurs dotés de 64 Go de mémoire. Le compilateur utilisé est gfortran avec la librairie MPI openmpi. Enfin les tests sont également faits sur **Beaufix**, un cluster BULL situé à Météo-France. Il est constitué de 1836 nœuds de calcul bi-socket Intel Broadwell, avec 40 cœurs par nœud doté d'une mémoire de 64 Go. Le compilateur utilisé pour faire les tests est intel avec la librairie Intel MPI impi.

Les commandes pour créer les quatre ***Buildslaves*** (esclaves) sont les suivantes :

- ***buildslave create-slave*** slave_napali_oa3-mct_tronc romulus.cerfacs.fr:9999 napali_oa3-mct_tronc build2017
- ***buildslave create-slave*** slave_neptune_oa3-mct_tronc romulus.cerfacs.fr:9999 neptune_oa3-mct_tronc build2017
- ***buildslave create-slave*** slave_nemoc_oa3-mct_tronc romulus.cerfacs.fr:9999 nemoc_oa3-mct_tronc build2017
- ***buildslave create-slave*** slave_beaufix_oa3-mct_tronc romulus.cerfacs.fr:9999 beaufix_oa3-mct_tronc build2017

Romulus est la machine Linux sur laquelle on fait tourner le ***Buildmaster*** (maître). Par exemple, pour la machine Napali, slave_napali_oa3-mct_tronc est le nom du répertoire associé au ***Buildslave*** (machine esclave) Napali qui s'appelle napali_oa3-mct_tronc. Le répertoire slave_napali_oa3-mct_tronc, contient les résultats des tests qui s'affichent sur internet (voir Figure 1). Build2017 est un mot de passe utilisé au moment de la définition du ***Buildslave*** (machine esclave) napali_oa3-mct_tronc. De même 9999 est le numéro du port TCP qui permet au ***Buildmaster*** (maître) d'écouter SVN et qui lui permet de communiquer avec les ***Buildslaves*** (machines esclaves). Ils sont définis dans la partie ***#BUILDSLAVES*** de ***master.cfg*** en Annexe A.

Les résultats de ces commandes sont donnés en Annexe B lorsque le lancement s'est bien passé.

2.3 Premier lancement de Buildbot, reconfiguration et arrêt

Une fois adapté le fichier de configuration ***master.cfg,*** il faut lancer le ***Buildmaster*** (maître) et les ***Buildslaves*** (esclaves) pour la première fois avec les commandes Linux :

- ***buildbot start*** master_oa3-mct_tronc
- ***buildslave start*** slave_napali_oa3-mct_tronc

- ***buildslave start*** slave_neptune_oa3-mct_tronc
- ***buildslave start*** slave_nemoc_oa3-mct_tronc
- ***buildslave start*** slave_beaufix_oa3-mct_tronc

Les résultats de ces commandes sont donnés en Annexe C lorsque le lancement s'est bien passé.

Si on fait une modification dans le fichier de configuration au cours des tests d'une version ou du tronc, par exemple si on ajoute un nouveau modèle jouet ou si on ajoute des tests sur un ancien modèle jouet, on peut les prendre en compte en utilisant la commande :

- ***buildbot reconfig*** master_oa3-mct_tronc

Pour arrêter définitivement ***Buildbot*** et tous les ***Buildslaves*** (machines esclaves), si on décide par exemple de tester une nouvelle configuration, on utilise les commandes Linux :

- ***buildbot stop*** master_oa3-mct_tronc
- ***buildslave stop*** slave_napali_oa3-mct_tronc
- ***buildslave stop*** slave_neptune_oa3-mct_tronc
- ***buildslave stop*** slave_nemoc_oa3-mct_tronc
- ***buildslave stop*** slave_beaufix_oa3-mct_tronc

Comme on le verra dans le paragraphe suivant, une fois que le processus ***Buildmaster*** (maître) et les processus ***Buildslaves*** (esclaves) tournent, on peut lancer et arrêter (mais pas définitivement) les ***Builders*** (tests) directement via l'interface graphique. Une autre façon de lancer des tests est de faire une modification du code sous SVN, comme on le verra dans la troisième partie.

2.4 Interface graphique de Buildbot

Les résultats sont visibles à l'adresse : http://romulus.cerfacs.fr:8015/waterfall . Cette adresse est définie dans la partie ***#PROJECT IDENTITY*** de ***master.cfg*** (voir Annexe A).

La Figure 2 ci-dessous montre ce que l'on obtient graphiquement, dans la fenêtre principale de ***Buildbot***, lors du premier lancement du ***Buildmaster*** (maître) et des ***Buildslaves*** (machines esclaves) via les commandes Linux définies dans la partie 2.3.

Dans le tableau des résultats à trois entrées, tests_beaufix_tronc, tests_napali_tronc, tests_nemoc_tronc et tests_neptune_tronc, définis en haut horizontalement, correspondent aux ***Builders*** (tests). Ils correspondent à la description exacte des tests à effectuer sur chaque ***Buildslave*** (machine esclave). La cellule « changes » correspond aux modifications du code sous SVN. La colonne de gauche du tableau correspond aux dates et heures auxquelles sont faits tous les ***Builders*** (tests) et toutes les modifications sous SVN.

En général les ***Builders*** (tests) se font après la mise à jour du code d'OASIS3-MCT sous SVN, mais il est toujours possible de déclencher les ***Builders*** (tests) et de les arrêter via l'interface graphique.

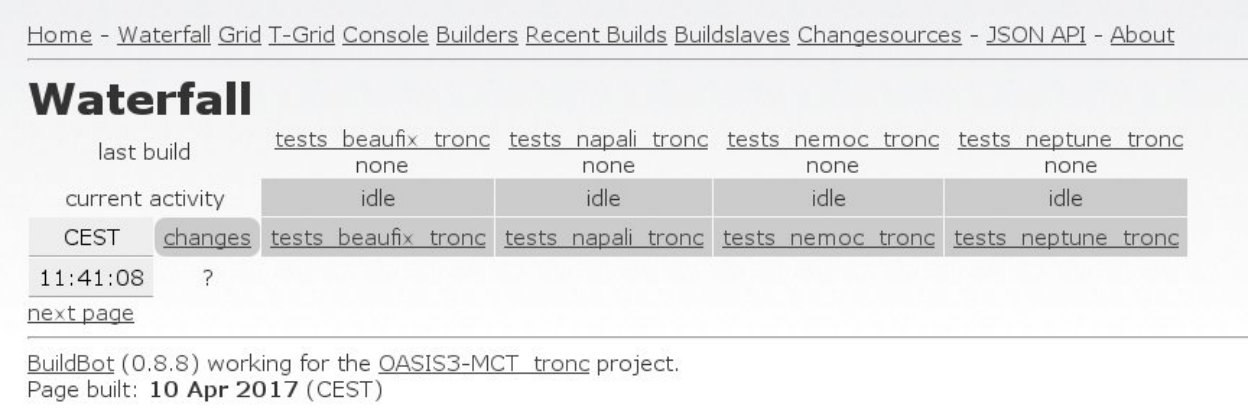


Figure 2 : Fenêtre principale de Buildbot au premier lancement

Si on clique par exemple sur tests_nemoc_tronc, Figure 2, on obtient la fenêtre représentée sur la Figure 3.1 ci-dessous à partir de laquelle on peut lancer un ***Build*** (ensemble des ***Builders*** sur un ***Buildslave***) en cliquant sur le bouton « Force Build ».

On obtient alors la Figure 3.2, où l'on peut voir qu'il y a maintenant un ***Current Builds*** en haut à gauche. Si on revient dans la fenêtre principale de ***Buildbot***, Figure 4.1, on voit que les calculs ont commencé sur le ***Buildslave*** Nemo_Lenovo.

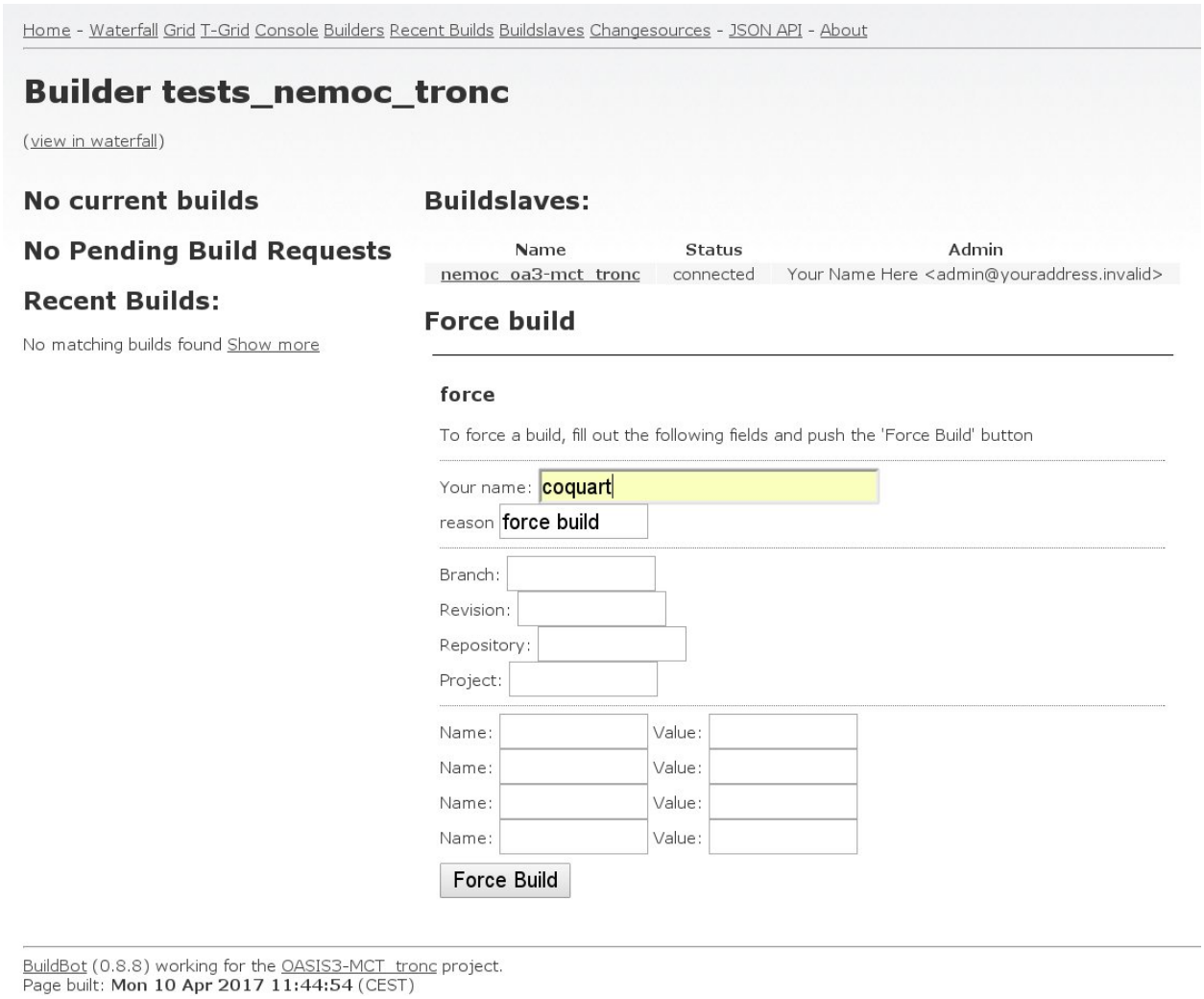


Figure 3.1 : Interface graphique de Buildbot : Force Build

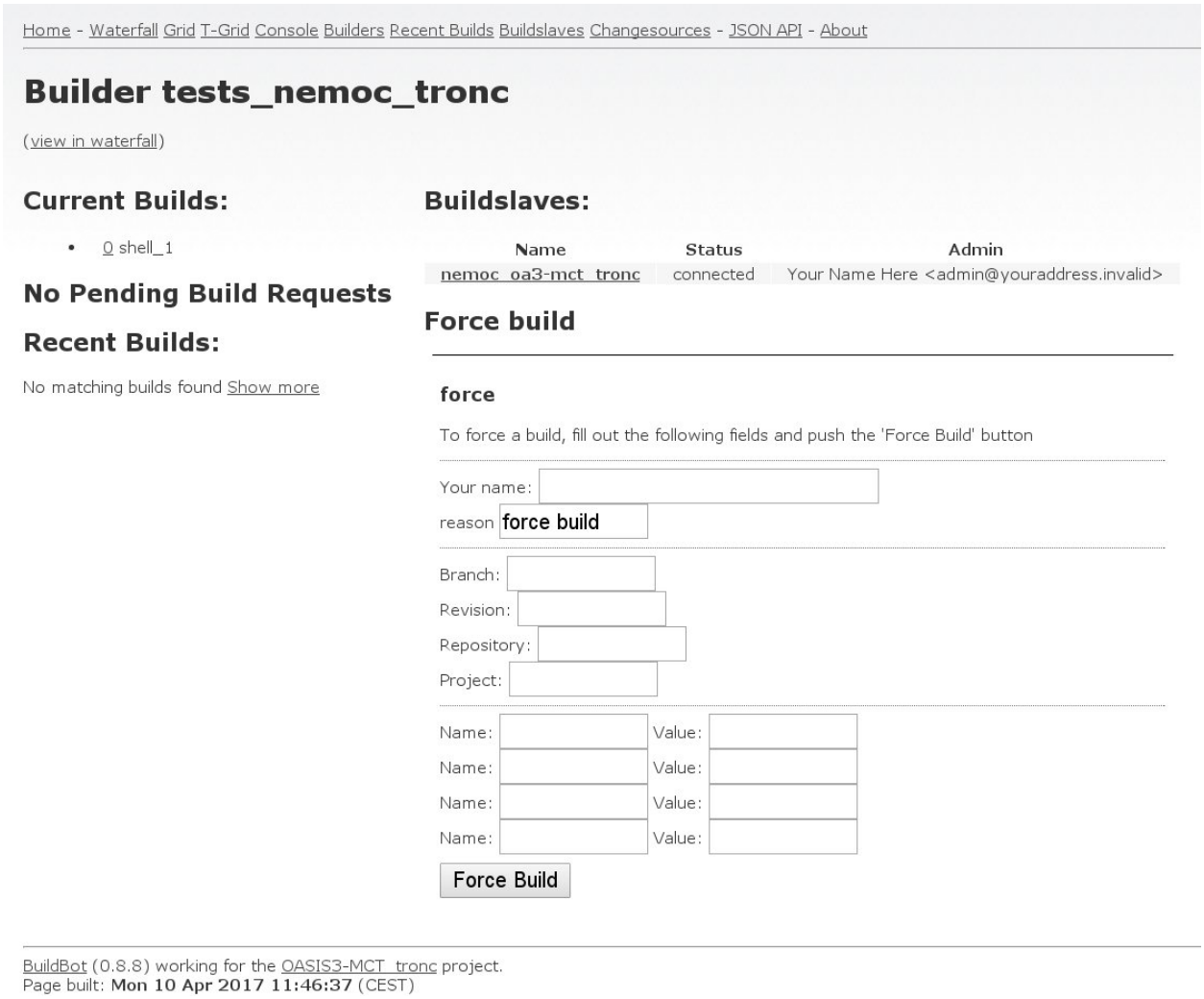


Figure 3.2 : Lancement du premier Build : Current Builds

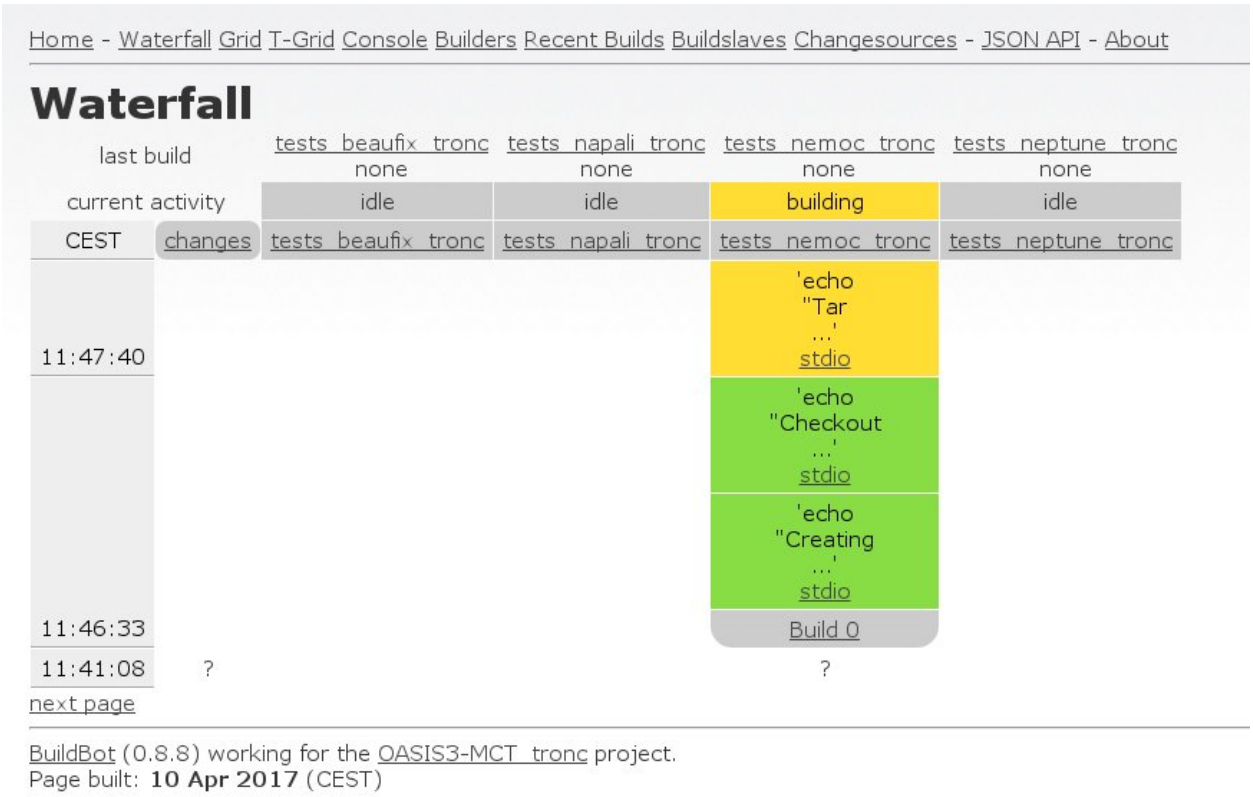


Figure 4.1 : Résultats du premier Build sur Nemo_Lenovo dans la fenêtre principale de Buildbot au cours du temps

Figure 4.2 : Résultats du premier Build sur Nemo_Lenovo dans la fenêtre principale de Buildbot au cours du temps

Dans la fenêtre principale représentée sur la Figure 4.2, un ***Builder*** (test) correspond à un rectangle, vert si tout s'est bien passé, jaune s'il est en train de tourner. On observe que le ***Buildmaster*** (maître) donne la date et l'heure auxquelles se produisent les différents ***Builder*** (test) dans la colonne de gauche.

Chaque ***Builder*** (test) sera explicité pour le tronc d'OASIS3-MCT dans la troisième partie.

Figure 5.1 : Arrêt manuel d'un Build sur Nemo_Lenovo via l'interface graphique de Buildbot

On peut également arrêter le ***Build*** sur Nemo_Lenovo via l'interface graphique en cliquant sur « Stop Build » comme le montre la Figure 5.1. Et on obtient alors l'arrêt des calculs comme le montrent les résultats obtenus dans la fenêtre principale de ***Buildbot***, Figure 5.2.

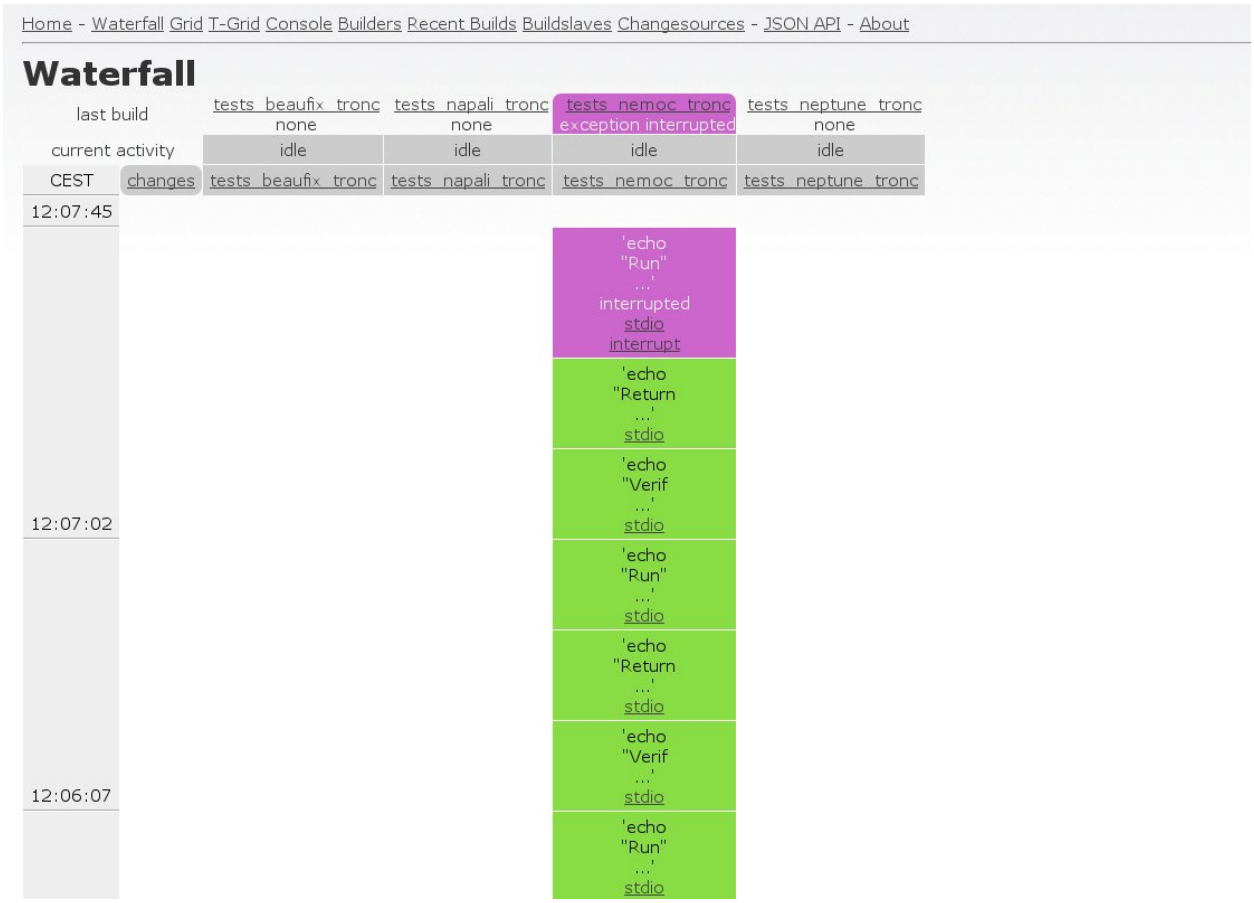


Figure 5.2 : Arrêt manuel des Builders (tests) sur Nemo_Lenovo dans la fenêtre principale

Comme on l'a déjà vu sur le Figure 4.2, un ***Builder*** (test) correspond à un rectangle, vert si tout s'est bien passé. La Figure 5.2 montre que le ***Buidler*** (test) est violet si on l'a arrêté via l'interface graphique (pas définitivement).

Dans la suite on verra que si un ***Builder*** (test) se passe mal, celui-ci apparaîtra en rouge et les commandes en cours s'arrêteront.

Pour ses codes de couleur, ***Buildbot*** se base sur le code de retour d'erreur du shell.

3 Résultats obtenus pour le tronc d'OASIS3-MCT

3.1 Description des tests

Pour ne pas alourdir le rapport, seules les premières commandes effectuées sur Nemo_Lenovo apparaissent dans le fichier de configuration ***master.cfg*** en Annexe A dans la partie ***#BUILDERS***, mais on a la même chose sur les trois autres ***Buildslaves*** (machines esclaves)

Si on regarde la structure du fichier ***master.cfg*** en Annexe A, chaque ***Builder*** (test) sur Nemo_Lenovo correspond en fait à une définition ***f1.addStep*** qui peut contenir une ou plusieurs commandes.

Le premier ***Builder*** « Create », comme indiqué sur la Figure 4.1, correspond à la création de tous les répertoires qui serviront pour les tests (voir f1addStep= « Create directories » dans ***master.cfg***). Le second ***Builder*** « Checkout » sur la Figure 4.1 correspond à l'extraction des sources du tronc d'OASIS3-MCT et des modèles jouets depuis SVN (voir f1addStep= «Checkout sources and examples of oasis3-mct" dans ***master.cfg***). Le troisième ***Builder*** « Tar » sur la Figure 4.1 correspond à la création des fichiers tar des sources d'OASIS3-MCT et des modèles jouets qui sont ensuite envoyés sur Nemo_Lenovo (voir f1addStep= « Tar and send files on nemoc » dans ***master.cfg***). On peut avoir des informations sur le résultat de chaque ***Builder*** (test) en cliquant sur le lien stdio dans l'interface graphique.

Sur la Figure 4.2, un autre ***Builder*** « Compile » compile alors les sources du tronc d'OASIS3-MCT sur Nemo_Lenovo (f1addStep= « Compile on nemoc » dans ***master.cfg***), puis d'autres ***Builders*** (tests) lancent successivement tous les tests pour chaque modèle jouet en monoprocesseur et en parallèle.

Seuls les ***Builders*** (tests) pour le premier test en monoprocesseur et en parallèle avec le modèle jouet test_simple_options sont donnés dans ***master.cfg*** :

*f1.addStep(ShellCommand(command='echo "Run" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/**run_examples_m2_oa3-mct_buildbot nemo_lenovo_gfortran_openmpi 0 1**', haltOnFailure=True, timeout=36000))*

*f1.addStep(ShellCommand(command='echo "Verif mono" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/**verif_files_m2_oa3-mct_buildbot nemo_lenovo_gfortran_openmpi 0 1**', haltOnFailure=True, timeout=36000))*

f1.addStep(ShellCommand(command='echo "Run" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-

*examples_buildbot/bench_buildbot/**run_examples_m2_oa3-mct_buildbot** **nemo_lenovo_gfortran_openmpi 0 1001**', haltOnFailure=True, timeout=36000))*

*f1.addStep(ShellCommand(command='echo "Verif para" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/**verif_files_m2_oa3-mct_buildbot** **nemo_lenovo_gfortran_openmpi 0 1001**', haltOnFailure=True, timeout=36000))*

Le ***Builder*** « Run » ci-dessus pour ce modèle jouet test_simple_options est basé sur le script ***run_examples_m2_oa3-mct_buildbot***, développé pour des modèles jouets avec deux composantes consistant en deux exécutables distincts. Le script lit sur quel ***Buildslave*** (machine esclave) il travaille (ici nemo_gfortran_openmpi), la précision du calcul (ici 0 qui correspond à un calcul en double précision), et enfin le test à effectuer, 1 en monoprocesseur ou 1001 en parallèle. Les chiffres 1 et 1001 correspondent aux fichiers de configuration du test casename1.txt et casename1001.txt, qui contiennent le nom du modèle jouet, le nom de la namcouple à tester (fichier de configuration d'OASIS3-MCT), le nom du Makefile à tester (car on peut tester différentes interpolations avec différentes décompositions), le nombre de processeurs pour chaque modèle (toujours 1 en monoprocesseur) et enfin les champs debug écrits par OASIS3-MCT à vérifier. Ces deux fichiers sont donc identiques, au nombre de processeurs près, et indépendants du ***Buildslave*** (esclave). On est donc sûr de faire à chaque fois les mêmes tests sur toutes les machines en changeant simplement l'argument correspondant au ***Buildslave*** (machine esclave).

Le script ***run_examples_m2_oa3-mct_buildbot*** avec l'argument ***nemo_lenovo_gfortran_openmpi*** permet de compiler puis de lancer les calculs sur Nemo_Lenovo, d'attendre la fin des calculs qui sont réalisés en batch, puis de créer un certain nombre de fichiers pour vérifier les résultats, en les comparant à un état de référence validé « à la main ».

Dans chaque répertoire de référence correspondant à chaque test validé, il y a un fichier, basé sur les fichiers de sortie d'OASIS3-MCT, permettant de vérifier que le calcul s'est bien terminé. Il y a un fichier avec les statistiques sur les champs de couplage aussi basé sur les fichiers de sortie d'OASIS3-MCT. Il y a un fichier contenant les minima et les maxima des champs de couplage calculés dans les modèles jouets. Il y a un fichier avec les statistiques en temps si cela a été demandé dans la namcouple. Il y a également un fichier par restart et un fichier par champ de couplage défini dans le fichier casename1.txt ou casename1001.txt et écrit par OASIS3-MCT (mode debug).

Dans le répertoire en cours d'analyse, il y a les mêmes fichiers avec à chaque fois, en plus, un fichier diff dans lequel se trouvent les différences entre les résultats du calcul en cours et les résultats de l'état de référence.

Dans le ***Builder*** suivant « Verif mono», qui utilise le script ***verif_files_m2_oa3-mct_buildbot***, la taille de chaque fichier diff est vérifiée et si elle n'est pas nulle le ***Builder*** « Verif mono » s'arrête et se retrouve en rouge dans l'interface graphique. Lorsque le calcul est parallèle, le ***Builder*** « Verif para » qui utilise aussi le script ***verif_files_m2_oa3-mct_buildbot***, permet de vérifier en plus qu'il n'y a pas de différence entre les champs de couplage écrit par OASIS3-MCT en monoprocesseur et ceux écrits en parallèle.

L'état de référence pour le tronc d'OASIS3-MCT contient 21 modèles jouets différents avec un, deux ou trois exécutables. Il s'est construit petit à petit depuis la version officielle OASIS3-MCT_1.0 car une partie des exemples restent les mêmes d'une version officielle à l'autre. Pour la version en développement du tronc d'OASIS3-MCT, il y a 292 ***Builders*** (un ***Build***) effectués avec les modèles jouets, qui sont réalisés sur 4 ***Buildslaves*** (machines esclaves) différents, soit **1168** tests réalisés à chaque fois qu'on lance ***Buildbot***.

Tous les tests pour le tronc d'OASIS3-MCT sont répertoriés sur la page wiki du CERFACS : https://inle.cerfacs.fr/projects/oasis3-mct/wiki/Toys_in_the_Buildbot_Suite_Tests_tronc

Cet outil a permis de trouver un certain nombre de bugs lors des développements du coupleur OASIS3-MCT.

3.2 Résultats graphiques

Une autre façon de déclencher les ***Builders*** (tests) consiste à faire une mise à jour des sources du tronc d'OASIS3-MCT sous SVN, sources accessibles à l'adresse (voir la partie ***#CHANGESOURCES*** dans le fichier ***master.cfg***) : https://oasis3mct.cerfacs.fr/svn/trunk/oasis3-mct

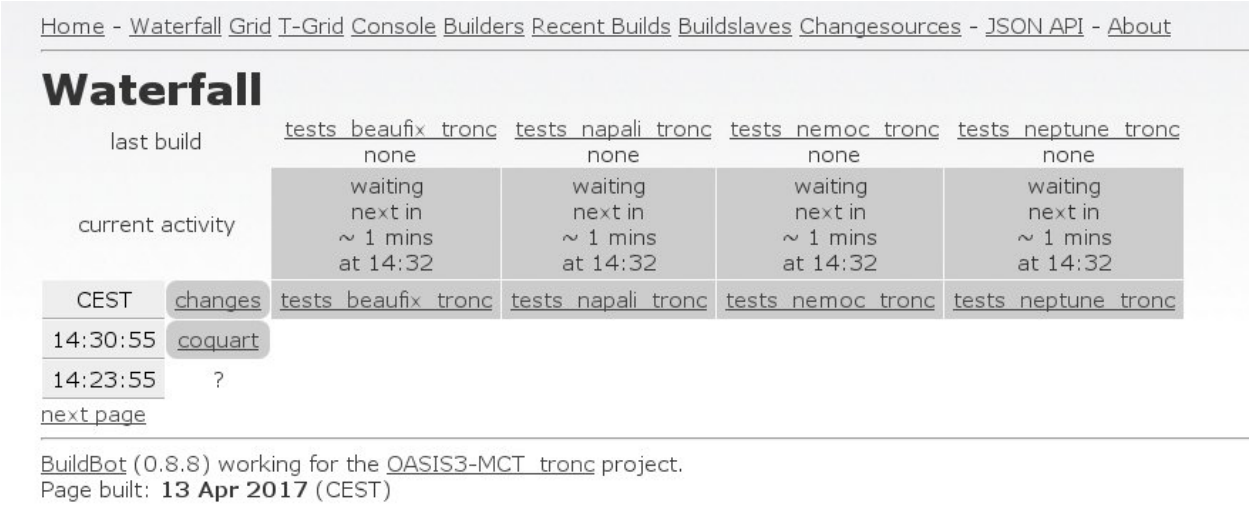


Figure 6.1 : Fenêtre principale de Buildbot au premier lancement et première modification sous SVN

Si un développeur fait une mise à jour dans les sources du tronc d'OASIS3-MCT sous SVN après le premier lancement de ***Buildbot***, on obtient la Figure 6.1. L'heure de la modification du code sous SVN apparaît dans la colonne de gauche et l'auteur de la modification apparait sous la cellule « Changes ». Les ***Builders*** tests_beaufix_tronc, tests_napali_tronc, tests_nemoc_tronc, tests_neptune_tronc indiquent sur la Figure 6.1 qu'ils sont en attente avant le lancement des ***Builds*** (ensembles des ***Builders*** sur les quatre ***Buildslaves***).

Dans notre cas, le ***Buildmaster*** (maître) écoute le serveur SVN toutes les 60 secondes pour voir s'il y a des modifications (voir pollinterval=60 dans la partie ***#CHANGESOURCES*** de ***master.cfg***) et les 120 secondes d'attente correspondent au temps d'attente des ***Schedulers*** (planificateurs) (voir treeStableTimer=120 dans la ***partie #SCHEDULERS*** de ***master.cfg***). Cette attente, paramétrable, permet d'avoir un code stable pour les tests.

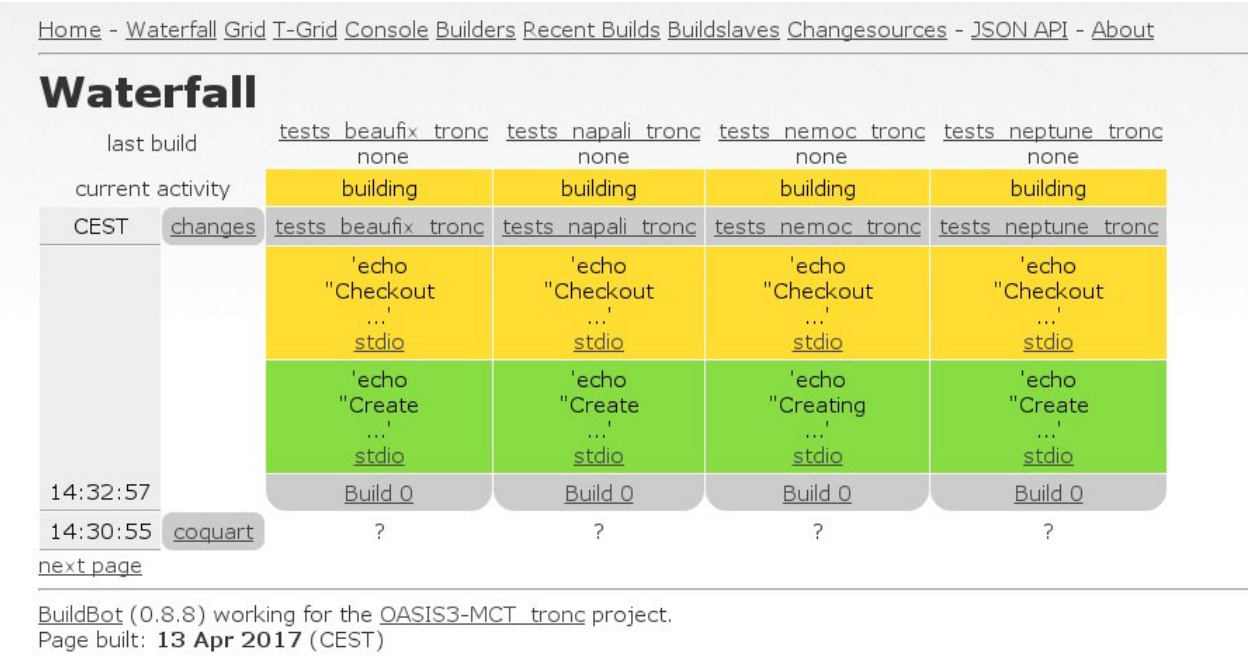
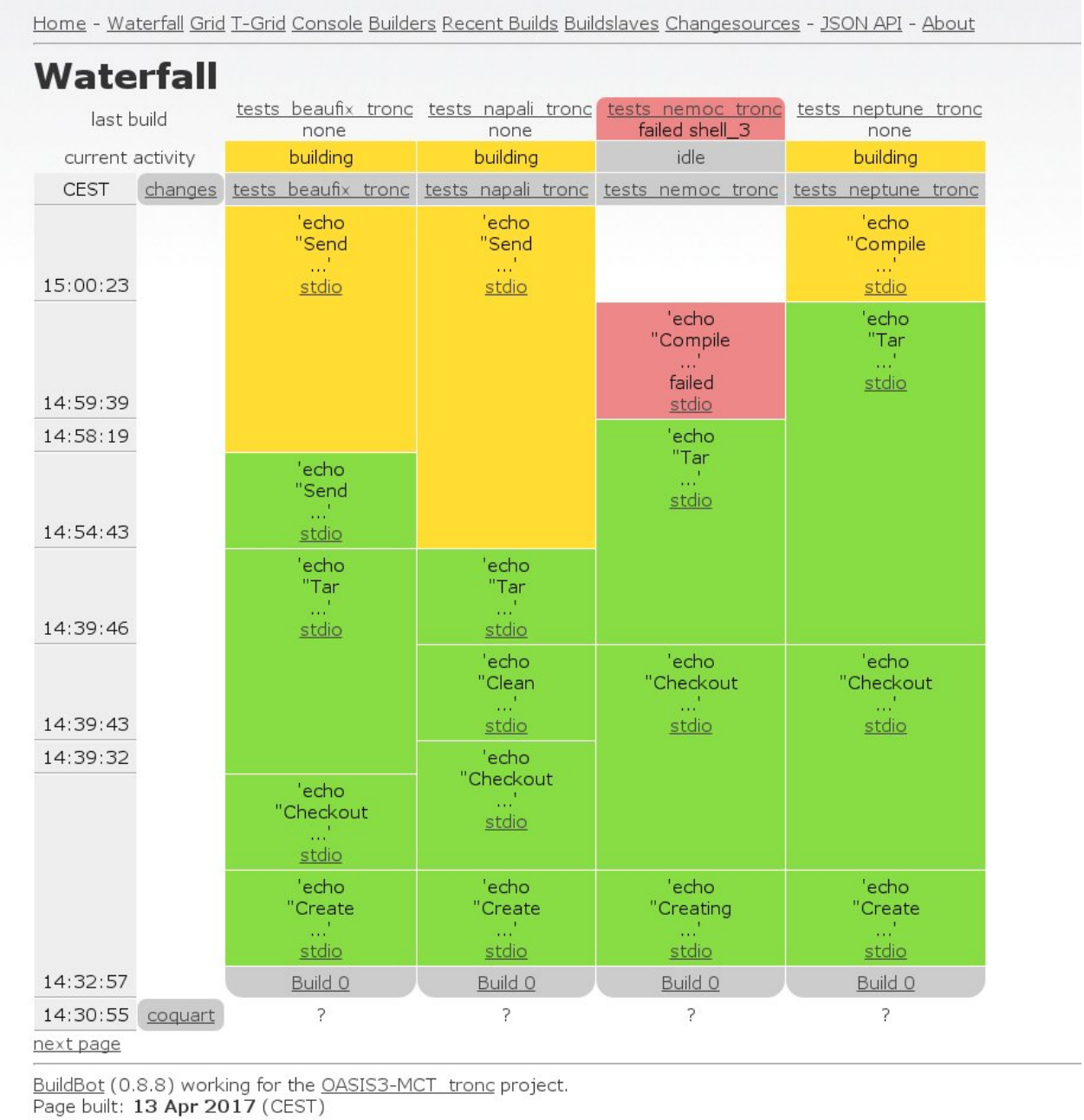


Figure 6.2 : Lancement automatique des Builds dans la fenêtre principale de Buildbot au bout de 120 secondes



Home - Waterfall Grid T-Grid Console Builders Recent Builds Buildslaves Changesources - JSON API - About

Waterfall

		tests_beaufix_tronc none	tests_napali_tronc none	tests_nemoc_tronc failed shell_3	tests_neptune_tronc none
last build					
current activity		building	building	idle	building
CEST	changes	tests_beaufix_tronc	tests_napali_tronc	tests_nemoc_tronc	tests_neptune_tronc
15:00:23		'echo "Send ...' stdio	'echo "Send ...' stdio		'echo "Compile ...' stdio
14:59:39				'echo "Compile ...' failed stdio	'echo "Tar ...' stdio
14:58:19				'echo "Tar ...' stdio	
14:54:43		'echo "Send ...' stdio			
14:39:46		'echo "Tar ...' stdio	'echo "Tar ...' stdio		
14:39:43			'echo "Clean ...' stdio	'echo "Checkout ...' stdio	'echo "Checkout ...' stdio
14:39:32			'echo "Checkout ...' stdio		
		'echo "Checkout ...' stdio			
		'echo "Create ...' stdio	'echo "Create ...' stdio	'echo "Creating ...' stdio	'echo "Create ...' stdio
14:32:57		Build 0	Build 0	Build 0	Build 0
14:30:55	coquart	?	?	?	?

next page

BuildBot (0.8.8) working for the OASIS3-MCT_tronc project.
Page built: **13 Apr 2017** (CEST)

Figure 6.3 : Arrêt du Builder « Compile » sur Nemo_Lenovo

La Figure 6.3 montre que la mise à jour des sources du tronc d'OASIS3-MCT sous SNV a introduit une erreur à la compilation, facilement visualisable par la couleur rouge du ***Builder*** « Compile ». Cette erreur se produit d'abord sur Nemo_Lenovo qui est en avance par rapport aux trois autres ***Buildslaves*** (machines esclaves).

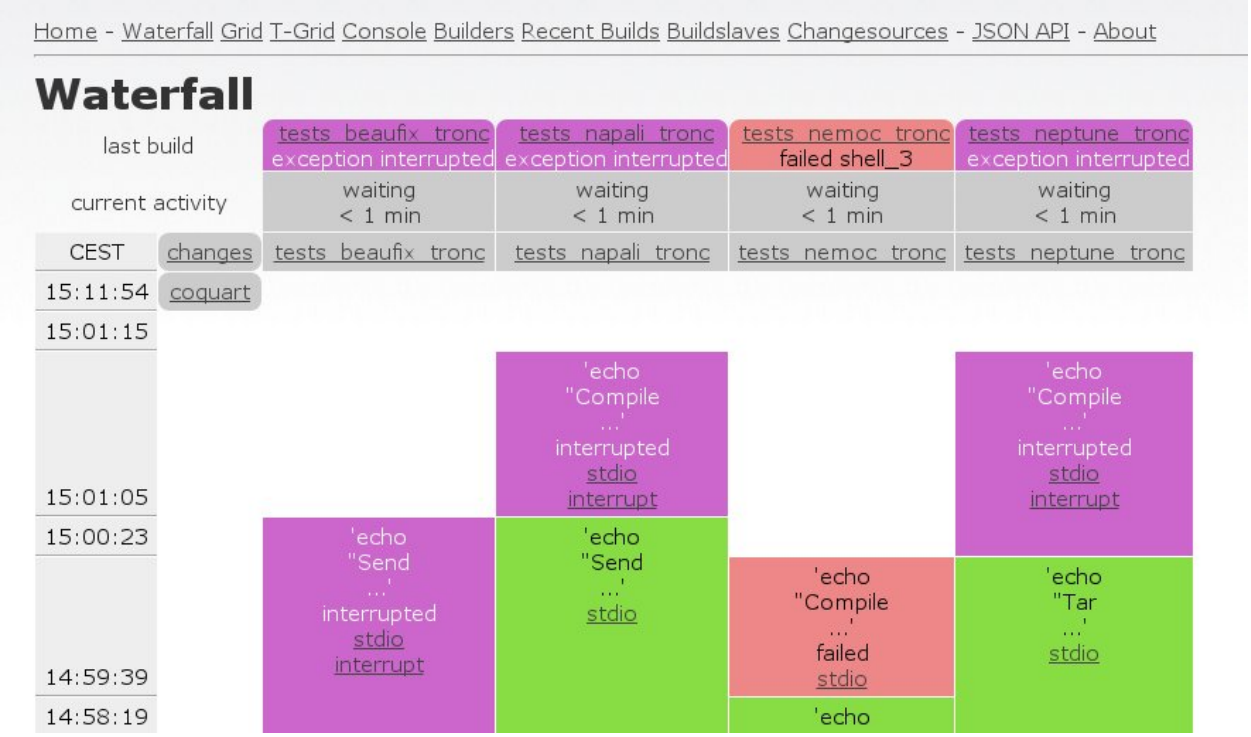


Figure 6.4 : Arrêt manuel des Builds sur Beaufix, Napali et Neptune et relance

Les trois autres ***Builds*** (ensembles des ***Builders*** sur les ***Buildslaves***) en cours sont arrêtés. Ils apparaissent alors en violet dans l'interface graphique, Figure 6.4.

Le bug est corrigé et les sources sont à nouveau mises à jour sous SVN. Les calculs vont se relancer au bout de 120 secondes (comme précédemment sur la Figure 6.2).

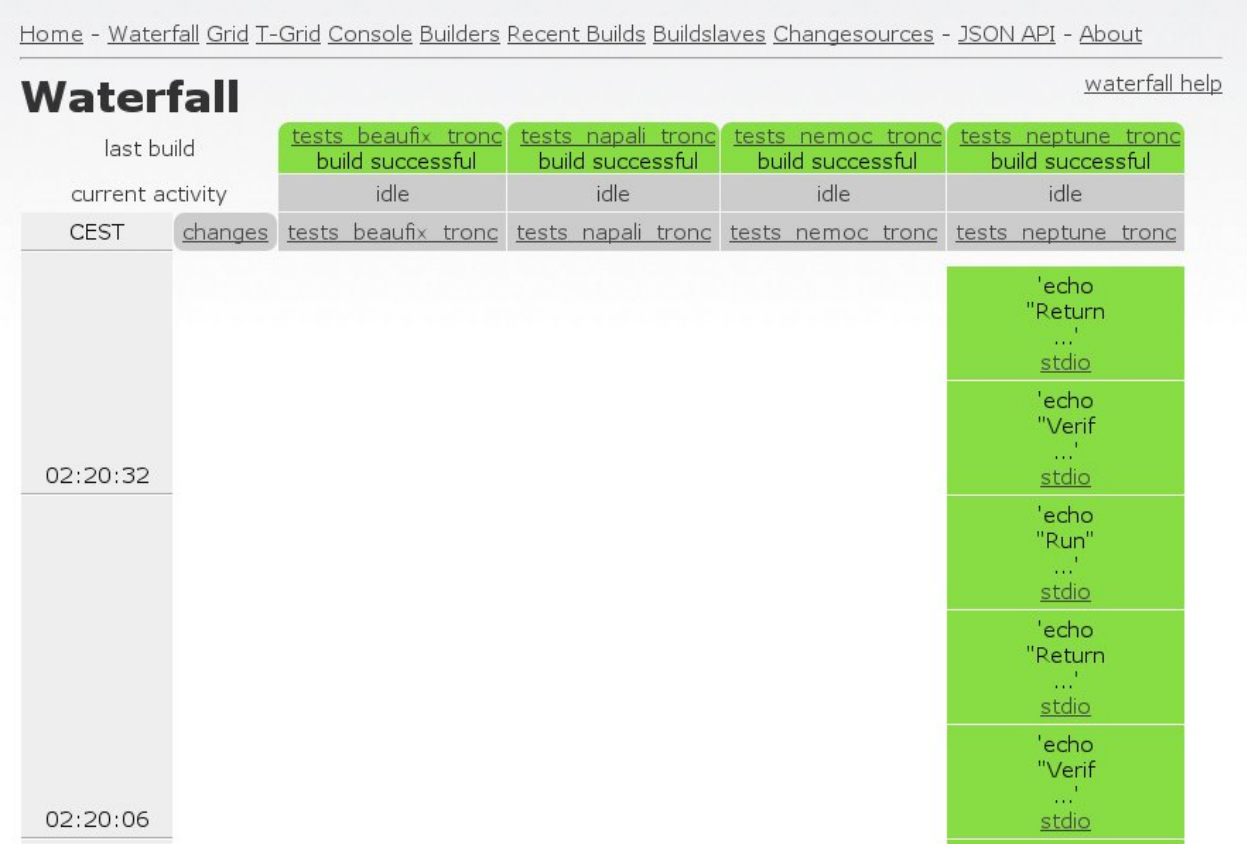


Figure 6.5 : Tests validés pour la révision en cours du tronc d'OASIS3-MCT dans la fenêtre principale de Buildbot

Cette fois ci tous les ***Builders*** (tests) sur tous les ***Buildslaves*** (esclaves) se sont bien passés et la révision en cours des sources du tronc d'OASIS3-MCT est validée.

4 Conclusion

Ce rapport présente le logiciel ***Buildbot***, utilisé pour tester en continu au cours du temps les sources du coupleur OASIS3-MCT développé au CERFACS/CNRS en collaboration avec Anthony Craig (UCAR, USA).

Les concepts de base puis le vocabulaire utilisé dans le fichier de configuration écrit en python, ***master.cfg***, sont détaillés dans (2.1), (2.2) et (2.3).

L'interface graphique, qui est un élément fondamental de ***Buildbot*** pour analyser facilement les résultats des nombreux tests effectués à chaque fois qu'une modification des sources du coupleur est envoyée sous SVN, est décrite en (2.4). Soit le test s'est bien passé et il apparaît en vert dans l'interface graphique, soit il s'est mal passé et il apparaît

en rouge dans l'interface graphique. Il peut aussi être violet si on l'a arrêté « à la main » via l'interface ou jaune s'il est en train de tourner.

La troisième et dernière partie, présente les tests (3.1) et les résultats graphiques (3.2) obtenus avec ***Buildbot*** qui sont plus spécifiques au développement des sources du tronc d'OASIS3-MCT. Une fois lancés les ***Builds*** (ensemble des ***Builders*** sur les ***Buildslaves***), soit via une modification sous SVN soit via l'interface graphique, les répertoires pour stocker les sources du tronc d'OASIS3-MCT et les sources des modèles jouets utilisés lors des tests sont créés. Puis les sources sont effectivement extraites via la commande ***svn checkout*** et sont envoyées sur chaque ***Buildslave*** (machine esclave). Les sources du coupleur sont compilées une seule fois sur chaque ***Buildslave*** (voir le ***Builder*** « Compile ») puis tous les ***Builders*** (tests) sont lancés successivement sur chaque ***Buildslave*** (machine esclave), en monoprocesseur puis en parallèle.

Il y a 292 ***Builders*** (un ***Build***) effectués avec 21 modèles jouets, réalisés sur 4 ***Buildslaves*** (machines esclaves) différents pour le tronc d'OASIS3-MCT, soit **1168** tests réalisés à chaque fois pour valider une révision du tronc d'OASIS3-MCT.

Les résultats des tests d'une révision en cours, sont comparés aux résultats d'un état de référence « validé à la main » et propre à chaque version officielle du coupleur. C'est ce qui permet de déceler les erreurs d'une mise à jour à l'autre sous SVN.

Cet outil est très simple d'utilisation, une fois construit le fichier de configuration ***master.cfg***, grâce à son interface graphique facilement interprétable et il a permis d'identifier un certain nombre de bugs lors des développements du coupleur OASIS3-MCT.

5 References

Buildbot Manual (2017), http://docs.buildbot.net/current/manual/index.html

Valcke, S., Craig, T. and Coquart, L. (2015), OASIS3-MCT User Guide, OASIS3-MCT3.0, URA SUC 1875, CERFACS/CNRS, TR-CMGC-15-22804, Toulouse, France , technical report

Valcke, S., Craig, T. and Coquart, L. (2013), OASIS3-MCT User Guide : OASIS3-MCT2.0, URA SUC 1875 CERFACS/CNRS, TR-CMGC-13-22805, Toulouse, France, technical report

Valcke, S., Craig, T. and Coquart, L. (2012), OASIS3-MCT User Guide, OASIS3-MCT1.0, URA SUC 1875, CERFACS/CNRS, TR-CMGC-12-22806, Toulouse, France , technical report

6 Annexe

6.1 Annexe A : Fichier python lu par le Buildmaster

```
# -*- python -*-

# ex: set syntax=python:

# This is a sample buildmaster config file. It must be installed as

# 'master.cfg' in your buildmaster's base directory.

# This is the dictionary that the buildmaster pays attention to. We also use

# a shorter alias to save typing.

c = BuildmasterConfig = {}

####### BUILDSLAVES

# The 'slaves' list defines the set of recognized buildslaves. Each element is

# a BuildSlave object, specifying a unique slave name and password.  The same

# slave name and password must be configured on the slave.

from buildbot.buildslave import BuildSlave

from buildbot.status     import mail

s1 = BuildSlave("nemoc_oa3-mct_tronc", "build2017",
notify_on_missing="coquart@cerfacs.fr")

s2 = BuildSlave("neptune_oa3-mct_tronc", "build2017",
notify_on_missing="coquart@cerfacs.fr")

s3 = BuildSlave("beaufix_oa3-mct_tronc", "build2017",
notify_on_missing="coquart@cerfacs.fr")

s4 = BuildSlave("napali_oa3-mct_tronc", "build2017",
notify_on_missing="coquart@cerfacs.fr")

c['slaves'] = [s1,s2,s3,s4]

# 'slavePortnum' defines the TCP port to listen on for connections from slaves,

# and SVN server.

# This must match the value configured into the buildslaves (with their

# --master option)

# Only one port for all slaves
```

```
c['slavePortnum'] = 9999
```

####### CHANGESOURCES

```
# the 'change_source' setting tells the buildmaster how it should find out
# about source code changes.
# it is possible to follow changes in multiple branches (trunk+branches) but
# not easy, need to split SVNPoller
# In my case, every Change that our SVNPoller produces will have .branch=None,
# to indicate that the Change is on the trunk. No other sub-projects or branches
# will be tracked.
from buildbot.changes.svnpoller import SVNPoller
svnroot = "https://oasis3mct.cerfacs.fr/svn/trunk/oasis3-mct"
c['change_source'] = []
c['change_source'] = []
c['change_source'] = []
c['change_source'].append(SVNPoller(svnurl=svnroot,
pollinterval=60,svnbin="/usr/bin/svn"))
```

####### SCHEDULERS

```
# Configure the Schedulers, which decide how to react to incoming changes.  In this
# case, just kick off a 'runtests' build
# branch=None means that the scheduler will track the trunk
from buildbot.schedulers.basic  import SingleBranchScheduler
from buildbot.schedulers  import basic
from buildbot.schedulers.forcesched import *
tests = SingleBranchScheduler(name="tests",branch=None,treeStableTimer=120,
builderNames=["tests_nemoc_tronc","tests_neptune_tronc","tests_beaufix_tronc","tes
ts_napali_tronc"])
```

```
c['schedulers'] = [tests]

sch = ForceScheduler(name="force",
builderNames=["tests_nemoc_tronc","tests_neptune_tronc","tests_beaufix_tronc","tes
ts_napali_tronc"])

c['schedulers'].append(sch)
```

####### BUILDERS

```
# The 'builders' list defines the Builders, which tell Buildbot how to perform a build:

# what steps, and which slaves can execute them.  Note that any particular build will

# only take place on one slave.

from buildbot.process import factory

from buildbot.steps.source import SVN

from buildbot.steps       import shell

from buildbot.steps.shell import ShellCommand

################################################################################

# NEMO_LENOVO : grfortran + openmpi pour un test sur test_simple_options

################################################################################

f1 = factory.BuildFactory()

# Launch sources and compile on nemo_lenovo account coquart

f1.addStep(ShellCommand(command='echo "Creating repositories" ; rm -rf
/space/coquart/nemoc ; mkdir /space/coquart/nemoc ; mkdir
/space/coquart/nemoc/oasis3-mct_buildbot ; mkdir /space/coquart/nemoc/oasis3-mct-
examples_buildbot ; mkdir /space/coquart/nemoc/oasis3-mct-
examples_buildbot/bench_buildbot ; mkdir /space/coquart/nemoc/oasis3-mct-
examples_buildbot/test_simple_options ;  mkdir /space/coquart/nemoc/oasis3-mct-
examples_buildbot/toy_eric_pulsation ; mkdir /space/coquart/nemoc/oasis3-mct-
examples_buildbot/toy_eric_echam_cosmo_cottbus ; mkdir
/space/coquart/nemoc/oasis3-mct-examples_buildbot/test_prism_oasis ; mkdir
/space/coquart/nemoc/oasis3-mct-examples_buildbot/mapchk ; mkdir
/space/coquart/nemoc/oasis3-mct-examples_buildbot/maphot ; mkdir
/space/coquart/nemoc/oasis3-mct-examples_buildbot/testr4r8 ; mkdir
/space/coquart/nemoc/oasis3-mct-examples_buildbot/test_gaussian_reduced ;  mkdir
```

/space/coquart/nemoc/oasis3-mct-examples_buildbot/test_identical_grids ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1fto2f_2models ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_3D_remap_file ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1bin_sequential ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1bin_ocnice ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/HR_tutorial ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_writing_grids ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/toy_auxiliary_routines ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_grouped_fields ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_bundles ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/tc3a ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/pes_coupling ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/no_pes_coupling ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_eric_3bin_io_2grids ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/toyhadgem3_UKCA ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/pes_coupling_3.0 ; mkdir /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1bin_concurrent', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Checkout sources and examples of oasis3-mct" ; cd /space/coquart/nemoc/oasis3-mct_buildbot ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/oasis3-mct . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/bench_buildbot ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/bench_buildbot . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_simple_options ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_simple_options . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/toy_eric_pulsation ; svn https://oasis3mct.cerfacs.fr/svn/trunk/examples/toy_eric_pulsation . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/toy_eric_echam_cosmo_cottbus ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples /toy_eric_echam_cosmo_cottbus . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_prism_oasis ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_prism_oasis . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/mapchk ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/mapchk . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/maphot ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/maphot . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/testr4r8 ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/testr4r8 . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_gaussian_reduced ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_gaussian_reduced . ; cd

/space/coquart/nemoc/oasis3-mct-examples_buildbot/test_identical_grids ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_identical_grids . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1fto2f_2models ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_1fto2f_2models . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_3D_remap_file ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_3D_remap_file . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1bin_sequential ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_1bin_sequential . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1bin_ocnice ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_1bin_ocnice . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/HR_tutorial ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/HR_tutorial . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_writing_grids ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_writing_grids . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/toy_auxiliary_routines ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/toy_auxiliary_routines . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_grouped_fields ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_grouped_fields . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_bundles ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_bundles . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/tc3a ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/tc3a . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/pes_coupling ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/pes_coupling . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/no_pes_coupling ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/no_pes_coupling . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_eric_3bin_io_2grids ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_eric_3bin_io_2grids . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/toyhadgem3_UKCA ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/toyhadgem3_UKCA . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/pes_coupling_3.0 ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/pes_coupling_3.0 . ; cd /space/coquart/nemoc/oasis3-mct-examples_buildbot/test_1bin_concurrent ; svn checkout https://oasis3mct.cerfacs.fr/svn/trunk/examples/test_1bin_concurrent .', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Tar and send files on nemoc" ; /space/coquart/nemoc/oasis3-mct-examples_buildbot/bench_buildbot/create_oasis3-mct_tar_platform_buildbot nemoc ; scp /space/coquart/nemoc/oasis3-mct_buildbot.tar coquart@nemo:/scratch/globc/coquart/OASIS3_BUILDBOT ; scp /space/coquart/nemoc/oasis3-mct-examples_buildbot.tar

coquart@nemo:/scratch/globc/coquart/OASIS3_BUILDBOT', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Compile on nemoc" ; ssh nemo /data/home/globc/coquart/bin/buildbot_compile_oasis3-mct_nemoc nemo_lenovo_gfortran_openmpi', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Clean sources and examples on romulus for nemoc" ; rm -rf /space/coquart/nemoc', haltOnFailure=True, timeout=36000))

launch tests on nemoc on mono

 # Toy test_simple_options

f1.addStep(ShellCommand(command='echo "Run" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/run_examples_m2_oa3-mct_buildbot nemo_lenovo_gfortran_openmpi 0 1', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Verif mono" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/verif_files_m2_oa3-mct_buildbot nemo_lenovo_gfortran_openmpi 0 1', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Return code" ; /wkdir/globc/coquart/BUILDBOT_OASIS/NEMO_LENOVO/end_of_verif_casename1', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Run" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/run_examples_m2_oa3-mct_buildbot nemo_lenovo_gfortran_openmpi 0 1001', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Verif para" ; ssh nemo /scratch/globc/coquart/OASIS3_BUILDBOT/oasis3-mct-examples_buildbot/bench_buildbot/verif_files_m2_oa3-mct_buildbot nemo_lenovo_gfortran_openmpi 0 1001', haltOnFailure=True, timeout=36000))

f1.addStep(ShellCommand(command='echo "Return code" ; /wkdir/globc/coquart/BUILDBOT_OASIS/NEMO_LENOVO/end_of_verif_casename1001', haltOnFailure=True, timeout=36000))

##

NEPTUNE : gfortran + impi : f2= factory.BuildFactory()

##

```
# BEAUFIX : intel + impi : f3= factory.BuildFactory()
################################################################################
# NAPALI : pgi + impi : f4= factory.BuildFactory()
################################################################################
b1 =  {'name': "tests_nemoc_tronc",
    'slavename': "nemoc_oa3-mct_tronc",
    'builddir': "subdir_slave_nemoc_oa3-mct_tronc",
    'factory': f1,
     }
b2 =  {'name': "tests_neptune_tronc",
    'slavename': "neptune_oa3-mct_tronc",
    'builddir': "subdir_slave_neptune_oa3-mct_tronc",
    'factory': f2,
    }
b3 =  {'name': "tests_beaufix_tronc",
    'slavename': "beaufix_oa3-mct_tronc",
    'builddir': "subdir_slave_beaufix_oa3-mct_tronc",
    'factory': f3,
    }
b4 =  {'name': "tests_napali_tronc",
    'slavename': "napali_oa3-mct_tronc",
    'builddir': "subdir_slave_napali_oa3-mct_tronc",
    'factory': f4,
    }
c['builders'] = [b1,b2,b3,b4]
```

####### STATUS TARGETS

```
# 'status' is a list of Status Targets. The results of each build will be
# pushed to these targets. buildbot/status/*.py has a variety to choose from,
# including web pages, email senders, and IRC bots.
# initial : from buildbot.status import html
#from buildbot.status.html import WebStatus
from buildbot.status import html
from buildbot.status.web.authz import Authz
authz_cfg = Authz(forceBuild=True, stopBuild=True)
c['status'] = []
c['status'].append(html.WebStatus(http_port=8015, authz=authz_cfg))
```

####### PROJECT IDENTITY

```
# the 'title' string will appear at the top of this buildbot
# installation's html.WebStatus home page (linked to the
# 'titleURL') and is embedded in the title of the waterfall HTML page.
c['title'] = "OASIS3-MCT_tronc"
c['titleURL'] = "https://verc.enes.org/oasis"
# the 'buildbotURL' string should point to the location where the buildbot's
# internal web server (usually the html.WebStatus page) is visible. This
# typically uses the port number set in the Waterfall 'status' entry, but
# with an externally-visible host name which the buildbot cannot figure out
# without some help.
c['buildbotURL'] = http://romulus.cerfacs.fr:8015/
```

####### DB URL

```
c['db'] = {
    # This specifies what database buildbot uses to store its state.  You can leave
```

```
    # this at its default for all but the largest installations.
    'db_url' : "sqlite:///state.sqlite",
}
```

6.2 Annexe B : Creation du Buildmaster et des Buildslaves

creating /space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/master.cfg.sample

populating public_html/

populating templates/

creating database (sqlite:///state.sqlite)

buildmaster configured in /space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_nemoc_oa3-mct_tronc

chdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_nemoc_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_nemoc_oa3-mct_tronc/info

Creating info/admin, you need to edit it appropriately

Creating info/host, you need to edit it appropriately

Not creating info/access_uri - add it if you wish

Please edit the files in /space/coquart/Buildbot/oasis3-mct_tronc/slave_nemoc_oa3-mct_tronc/info appropriately.

buildslave configured in /space/coquart/Buildbot/oasis3-mct_tronc/slave_nemoc_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_neptune_oa3-mct_tronc

chdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_neptune_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_neptune_oa3-mct_tronc/info

Creating info/admin, you need to edit it appropriately

Creating info/host, you need to edit it appropriately

Not creating info/access_uri - add it if you wish

Please edit the files in /space/coquart/Buildbot/oasis3-mct_tronc/slave_neptune_oa3-mct_tronc/info appropriately.

buildslave configured in /space/coquart/Buildbot/oasis3-mct_tronc/slave_neptune_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_beaufix_oa3-mct_tronc

chdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_beaufix_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_beaufix_oa3-mct_tronc/info

Creating info/admin, you need to edit it appropriately

Creating info/host, you need to edit it appropriately

Not creating info/access_uri - add it if you wish

Please edit the files in /space/coquart/Buildbot/oasis3-mct_tronc/slave_beaufix_oa3-mct_tronc/info appropriately.

buildslave configured in /space/coquart/Buildbot/oasis3-mct_tronc/slave_beaufix_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_napali_oa3-mct_tronc

chdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_napali_oa3-mct_tronc

mkdir /space/coquart/Buildbot/oasis3-mct_tronc/slave_napali_oa3-mct_tronc/info

Creating info/admin, you need to edit it appropriately

Creating info/host, you need to edit it appropriately

Not creating info/access_uri - add it if you wish

Please edit the files in /space/coquart/Buildbot/oasis3-mct_tronc/slave_napali_oa3-mct_tronc/info appropriately.

buildslave configured in /space/coquart/Buildbot/oasis3-mct_tronc/slave_napali_oa3-mct_tronc

6.3 Annexe C : Lancement du Buildmaster et des Buildslaves

Following twistd.log until startup finished..

2017-04-03 14:24:34+0200 [-] Log opened.

2017-04-03 14:24:34+0200 [-] twistd 12.2.0 (/usr/bin/python 2.7.5) starting up.

2017-04-03 14:24:34+0200 [-] reactor class: twisted.internet.epollreactor.EPollReactor.

2017-04-03 14:24:34+0200 [-] Starting BuildMaster -- buildbot.version: 0.8.8

2017-04-03 14:24:34+0200 [-] Loading configuration from '/space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/master.cfg'

2017-04-03 14:24:34+0200 [-] Setting up database with URL 'sqlite:///state.sqlite'

2017-04-03 14:24:34+0200 [-] setting database journal mode to 'wal'

2017-04-03 14:24:35+0200 [-] adding 1 new changesources, removing 0

2017-04-03 14:24:35+0200 [-] adding 4 new slaves, removing 0

2017-04-03 14:24:35+0200 [-] adding 4 new builders, removing 0

2017-04-03 14:24:35+0200 [-] trying to load status pickle from /space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/subdir_slave_beaufix_oa3-mct_tronc/builder

2017-04-03 14:24:35+0200 [-] added builder tests_beaufix_tronc in category

2017-04-03 14:24:35+0200 [-] trying to load status pickle from /space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/subdir_slave_nemoc_oa3-mct_tronc/builder

2017-04-03 14:24:35+0200 [-] added builder tests_nemoc_tronc in category

2017-04-03 14:24:35+0200 [-] trying to load status pickle from /space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/subdir_slave_napali_oa3-mct_tronc/builder

2017-04-03 14:24:35+0200 [-] added builder tests_napali_tronc in category

2017-04-03 14:24:35+0200 [-] trying to load status pickle from /space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/subdir_slave_neptune_oa3-mct_tronc/builder

2017-04-03 14:24:35+0200 [-] added builder tests_neptune_tronc in category

2017-04-03 14:24:35+0200 [-] PBServerFactory starting on 9999

2017-04-03 14:24:35+0200 [-] Starting factory <twisted.spread.pb.PBServerFactory instance at 0x4678cb0>

2017-04-03 14:24:35+0200 [-] adding scheduler 'tests'

2017-04-03 14:24:35+0200 [-] adding scheduler 'force'

2017-04-03 14:24:35+0200 [-] WebStatus using (/space/coquart/Buildbot/oasis3-mct_tronc/master_oa3-mct_tronc/public_html)

2017-04-03 14:24:35+0200 [-] RotateLogSite starting on 8015

2017-04-03 14:24:35+0200 [-] Starting factory <buildbot.status.web.baseweb.RotateLogSite instance at 0x468dab8>

2017-04-03 14:24:35+0200 [-] Setting up http.log rotating 10 files of 10000000 bytes each

2017-04-03 14:24:35+0200 [-] BuildMaster is running

The buildmaster appears to have (re)started correctly.

Following twistd.log until startup finished..

2017-04-03 14:24:35+0200 [-] Log opened.

2017-04-03 14:24:35+0200 [-] twistd 12.2.0 (/usr/bin/python 2.7.5) starting up.

2017-04-03 14:24:35+0200 [-] reactor class: twisted.internet.epollreactor.EPollReactor.

2017-04-03 14:24:35+0200 [-] Starting BuildSlave -- version: 0.8.8

2017-04-03 14:24:35+0200 [-] recording hostname in twistd.hostname

2017-04-03 14:24:35+0200 [-] Starting factory <buildslave.bot.BotFactory instance at 0x147e830>

2017-04-03 14:24:35+0200 [-] Connecting to 127.0.0.1:9999

2017-04-03 14:24:35+0200 [Broker,client] message from master: attached

The buildslave appears to have (re)started correctly.

Following twistd.log until startup finished..

2017-04-03 14:24:35+0200 [-] Log opened.

2017-04-03 14:24:35+0200 [-] twistd 12.2.0 (/usr/bin/python 2.7.5) starting up.

2017-04-03 14:24:35+0200 [-] reactor class: twisted.internet.epollreactor.EPollReactor.

2017-04-03 14:24:35+0200 [-] Starting BuildSlave -- version: 0.8.8

2017-04-03 14:24:35+0200 [-] recording hostname in twistd.hostname

2017-04-03 14:24:35+0200 [-] Starting factory <buildslave.bot.BotFactory instance at 0x195c830>

2017-04-03 14:24:35+0200 [-] Connecting to 127.0.0.1:9999

2017-04-03 14:24:35+0200 [Broker,client] message from master: attached

The buildslave appears to have (re)started correctly.

Following twistd.log until startup finished..

2017-04-03 14:24:36+0200 [-] Log opened.

2017-04-03 14:24:36+0200 [-] twistd 12.2.0 (/usr/bin/python 2.7.5) starting up.

2017-04-03 14:24:36+0200 [-] reactor class: twisted.internet.epollreactor.EPollReactor.

2017-04-03 14:24:36+0200 [-] Starting BuildSlave -- version: 0.8.8

2017-04-03 14:24:36+0200 [-] recording hostname in twistd.hostname

2017-04-03 14:24:36+0200 [-] Starting factory <buildslave.bot.BotFactory instance at 0x1b99830>

2017-04-03 14:24:36+0200 [-] Connecting to 127.0.0.1:9999

2017-04-03 14:24:36+0200 [Broker,client] message from master: attached

The buildslave appears to have (re)started correctly.

Following twistd.log until startup finished..

2017-04-03 14:24:36+0200 [-] Log opened.

2017-04-03 14:24:36+0200 [-] twistd 12.2.0 (/usr/bin/python 2.7.5) starting up.

2017-04-03 14:24:36+0200 [-] reactor class: twisted.internet.epollreactor.EPollReactor.

2017-04-03 14:24:36+0200 [-] Starting BuildSlave -- version: 0.8.8

2017-04-03 14:24:36+0200 [-] recording hostname in twistd.hostname

2017-04-03 14:24:36+0200 [-] Starting factory <buildslave.bot.BotFactory instance at 0x2733830>

2017-04-03 14:24:36+0200 [-] Connecting to 127.0.0.1:9999

2017-04-03 14:24:36+0200 [Broker,client] message from master: attached

The buildslave appears to have (re)started correctly.