Jenkins technical documentation (Management)

Server setup

- Navigate to http://master_hostname:8080
- Navigate to Manage Jenkins -> Configure System and provide System Admin e-mail address . For this step and all that follows remember to save configuration on each page!
- Navigate to Manage Jenkins -> Manage Plugins and apply all available updates for installed plugins. Click All at the bottom of the page and then Download now and install after restart button. Install also Pre SCM BuildStep Plugin and Git plugin by navigating to Available tab, selecting the plugin and clicking Download now and install after restart button. You can easily find the plugin by typing its short name preSCMbuildstep and git in the Filter textbox in the upper right corner of the page. Remember to select Restart Jenkins when installation is complete and no jobs are running checkbox.
- Navigate to Manage Jenkins -> Configure System and in git section set Path to Git executable to /usr/local/bin/git .
- Navigate to Manage Jenkins -> Configure Global Security :
    - Select Enable security
    - Disable TCP port for JNLP slave agents
    - In Access Control
        - For Security Realm select Jenkins own user database and uncheck "Allow users to sign up".
        - For Authorization temporarily select Anyone can do anything.
    - Select Prevent Cross Site Request Forgery exploits and for Crumbs Default Crumb Issuer along with Enable proxy compatibility
    - After save a signup page should show up where administrator account should be created. It is recommended to call this user admin .
- Navigate to Manage Jenkins -> Configure Global Security and in Access Control for Authorization select Matrix-based security and add admin user, created in the previous step, by providing his name and granting him all rights. Revoke all right for anonymous users.
- Navigate to Manage Jenkins -> Manage Nodes .
    - Click on the master node and than on the Configure button. Set number of executors to 0 (as mentioned previously master node should not execute any jobs).
    - Click New Node button, provide node name, select Dumb Slave checkbox and click OK . For next slaves you can select Copy Existing Node option which allows you to use other agent configuration.
    - Provide:
        - Slave description
        - Number of executors (should be close or equal to the number of machine s cores)
        - Remote root directory as /var/jenkins. If you want to use other directory, make sure agent has read and write access for this directory.
        - Appropriate Labels , e.g. SLC6 so that this agent will build plans that should be built on the SLC6 OS.
        - For Usage choose Only build jobs with labels restrictions matching this node .
        - For launch method choose Launch slave agents on Unix machines via SSH and provide fully qualified slave s machine hostname. (In order to find out the fully qualified hostname of a machine log in to it and execute in the terminal hostname -f command.) Next select credentials or add one. In case you are creating new credentials:
            - If you base master-slave communication on public key infrastructure choose kind as SSH Username with private key , provide agent name (e.g. totemjenkins) and select Private key from the Jenkins master ~/.ssh . Provide password if private key is protected.

        - If you base master-slave communication on username/password choose kind as Username with password and provide username (e.g. totemjenkins) and password to this account (same that has been used in the previous section to login from master to slave machine).
    - ◊ For availability select Keep this slave online as much as possible.
- Navigate to Manage Jenkins -> Configure Global Security and select Enable Slave -> Master Access Control checkbox.

Plan creation

- At the homepage click on the New Item button.
    - Provide plan name. Plan name should not contain whitespace characters.
    - Choose Freestyle project checkbox. As mentioned previously it is useful to choose Copy existing Item in order to utilize existing configurations.
- Select Discard Old Builds and provide Max # of builds to keep .
- Select "This build is parameterized" checkbox and add "Boolean Parameter" with following properties:
    - Name: FORCE_CLEAN_BUILD
    - Default Value: checked
    - Description: Removes all files from the workspace before the build. It ensures that the workspace is in the pristine state.
- You can restrict where given plan can be build by selecting Restrict where this project can be run and providing Label expression .
- Select subversion as Source Code Management and provide
    - Repository URL, e.g. svn+ssh://svn.cern.ch/reps/totem/branches/CMSSW_7_0_4/offline/cmssw/src.
    - Credentials as username and password of user that has access to the repository. This should be a dedicated user, that only has rights to checkout the source code, instead of your own user. Use for instance the totemjenkins user.
    - Local module directory. It is a directory in which source code will be downloaded. For instance for CMSSW project it should be ./${JOB_NAME}/src .
    - For repository depth as-it-is .
    - Make sure ignore externals checkbox is unchecked.
    - Check-out Strategy should be Always checkout a fresh copy.
    - Repository browser should be Auto
- Select Build periodically in Build Triggers section and provide schedule expression e.g. H H(0-2) * * * in order to run plan some random time between 0 and 2 AM
- In Build Environment section select Run buildstep before SCM runs and provide Execute shell script as:

```
if $FORCE_CLEAN_BUILD; then
set -e # Exit immediately if a simple command exits with a non-zero status.
set -x # Log all commands to stdout.
set -o pipefail # Return value of a pipeline as the value of the last command to
                # exit with a non-zero status, or zero if all commands in the
                # pipeline exit successfully.

        AGENT_TEMP_LOG=`mktemp`
        AGENT_BUILD_LOG="$WORKSPACE/build.log"

function prepare_workspace() {
echo "Preparing workspace directory '$WORKSPACE'..."
cd "$WORKSPACE"
                rm -rf *
echo "Content of workspace directory '`pwd`':"
                ls -al
echo "File system disk space usage:"
                df -h
        }

        prepare_workspace 2>&1 | tee -a "$AGENT_TEMP_LOG"
    mv "$AGENT_TEMP_LOG" "$AGENT_BUILD_LOG"
fi
```

Remember to select Fail the build on error checkbox.

- Add build steps as Execute shell . Sample script building CMSSW_7_0_4 project using scripts included in the section below.

```
./"$JOB_NAME"/src/build.sh "buildMergingSoftware"
```

- Add another build steps as Execute shell .

```
./"$JOB_NAME"/src/test.sh
```

- For Post-build Actions choose:
    - ♦ E-mail notification
        - ◊ Provide recipient email address.
        - ◊ Select Send email for each unstable build checkbox.
    - ♦ Archive the artifacts
        - ◊ Files to archive pattern should be build.log .

Jenkins dependent plans

There is a possibility to create dependent plans or to split a complex plan into a series of dependent plans. Below you can find steps required to decouple build phase from test phase for CMSSW_7_0_4 plan.

- Create a copy of plan for CMSSW_7_0_4 by clicking to "New Item" button on the main page, selecting "Copy existing Item" checkbox and choosing CMSSW_7_0_4 plan. Provide new plan name, e.g. "CMSSW_7_0_4_tests".
    - ♦ Remove all parameters in "This build is parameterized" section and add one "String Parameter" with following properties:
        - ◊ Name: JOB_NAME
        - ◊ Default Value: CMSSW_7_0_4
        - ◊ Description: Name of parent job.
    - ♦ In "Advanced Project Options" select "Block build when upstream project is building" and "Use custom workspace" checkboxes. Custom workspace directory set to: "workspace/$JOB_NAME".
    - ♦ For "Source Code Management" choose "None".
    - ♦ In "Build Triggers" section uncheck "Build periodically" and check "Build after other projects are built" . For "Projects to watch" type "CMSSW_7_0_4" and select "Trigger only if build is stable".
    - ♦ Uncheck "Run buildstep before SCM runs" checkbox.
    - ♦ Remove "Execute shell" step responsible for building CMSSW project in "Build" section.
    - ♦ For "Files to archive" in "Post-build Actions" section provide: "test.log, *.root".
- Navigate to CMSSW_7_0_4 plan configuration "Plan name" -> "Configure". In "Advanced Project Options" section select "Block build when downstream project is building" checkbox.

Building scripts sample for CMSSW_7_0_4 project

```
#! /bin/bash

# Script used by Jenkins continuous integration server in order to build the
# project.
```

```
AGENT_USERNAME=`whoami`
AGENT_KERBEROS_KEYTAB="/etc/$AGENT_USERNAME.keytab"
AGENT_EXECUTORS=4
AGENT_BUILD_LOG="$WORKSPACE/build.log"

set -e # Exit immediately if a simple command exits with a non-zero status.
set -x # Log all commands to stdout.
set -o pipefail # Return value of a pipeline as the value of the last command to
                # exit with a non-zero status, or zero if all commands in the
                # pipeline exit successfully.
shopt -s expand_aliases # Expand command alias to the command itself.
                        # Required for non-interactive shell.
source /afs/cern.ch/cms/cmsset_default.sh

# Shows directory details.
function show_current_directory_details() {
echo "Content of directory '`pwd`':"
        ls -al
}

# Initializes Kerberos keytab for agent user.
function initialize_kerberos_keytab() {
echo "Initializing Kerberos keytab for user '$AGENT_USERNAME'..."
        kinit -kt "$AGENT_KERBEROS_KEYTAB" "$AGENT_USERNAME"
}

# Initializes scram project.
# $1 - project name
# $2 - project version
function initialize_scram_project() {
echo "Initializing scram project..."
        show_current_directory_details
        scram project "$JOB_NAME" "$1" "$2"
        show_current_directory_details
cd "$JOB_NAME"
        show_current_directory_details
        cmsenv
}

# Compiles scram project using AGENT_EXECUTORS number of threads.
function compile_scram_project() {
echo "Compiling scram project..."
cd src
        show_current_directory_details
echo "Starting parallel compilation using up to $AGENT_EXECUTORS threads..."
        scram build -j "$AGENT_EXECUTORS" || \
        scram build -j "$AGENT_EXECUTORS" || \
echo "Parallel compilation using up to $AGENT_EXECUTORS threads failed..."
echo "Starting sequential compilation..."
        scram build
}

# Checks whether merging software build step is required and proceeds with the
# build if it is.
function maybe_build_merging_software() {
if [ "XbuildMergingSoftware" = "X$1" ]; then
echo "Building merging software..."
              cd MergingSoftware/MergeCMSTOTEMNTuples/Merge/
                make -j "$AGENT_EXECUTORS"
fi
}

# Builds CMSSW project, version 7.0.4.
# $1 - if argument equals "buildMergingSoftware" an additional build step will
#      be executed
function build_CMSSW_7_0_4() {
```

```
echo "Building CMSSW 7.0.4 project..."
        initialize_kerberos_keytab
        initialize_scram_project "CMSSW" "CMSSW_7_0_4"
        compile_scram_project
        maybe_build_merging_software "$1"
}

# Executes given command and logs stdout both to the file and the screen.
# $1 - commnad to be executed
function execute_and_log() {
echo "Executing command '$1'"
eval "$1" 2>&1 | tee -a "$AGENT_BUILD_LOG"
}

execute_and_log "build_CMSSW_7_0_4 $1"
```

This topic: TOTEM > JenkinsTechDocuMgmt
Topic revision: r2 - 2015-09-23 - KrzysztofAndrzejTrzepla