

# CS 427

## Final Documentation

Name: Andrew Borg, Thomas Ciemniak, Rohan Kapoor,  
Kevin Ly, Guoqiao Li, Yang Sang, Stephen Sullivan

December 9, 2015

## Table of Contents

<b>1</b>	<b>Project Description</b>	<b>3</b>
1.1	Goals . . . . .	3
<b>2</b>	<b>Architecture</b>	<b>3</b>
<b>3</b>	<b>Design</b>	<b>6</b>
<b>4</b>	<b>Usage</b>	<b>7</b>

## List of Figures

1	Get the HPI needed to install the plugin by going into trunk/jobConfigHistory-plugin and typing "mvn compile && mvn hpi:hpi" . . . . .	7
2	Go into your Jenkins instance and go to Manage Jenkins->Manage Plugins . . . . .	8
3	Get the hpi file by going into /trunk/jobConfigHistory-plugin/target	9
4	Go back to the Manage Jenkins page and click Configure System	9
5	Enter the details of the SVN Repository You Want To Sync To	10
6	Go to Your Jenkins Home Page and Click on the JobConfigHistory Tab and enter a commit message/tag . . . . .	10
7	Go to your SVN URL and see that the commit message was saved . . . . .	11
8	Go to a Project and go to the Configuration Tab and Make a Change . . . . .	11
9	As you can see, a change was made . . . . .	12
10	Go back to the Jenkins home page and go to the Job Config History Tab->Show all configs and click on Yes for a configuration that you want to rollback to . . . . .	12

# 1 Project Description

As someone who uses Jenkins every day in the workplace, Rohan has a severe problem of keeping track of Jenkins Configurations, who edited job settings and why they were changed. The more jobs that a single Jenkins instance is keeping track of, the more of a problem this becomes. For reference, where Rohan works, a single Jenkins instance is used by 30 employees and has around 50 jobs on it across three programming languages. As one of the people responsible for maintaining the Jenkins configuration, Rohan wanted a nice way to store the configuration separate from the Jenkins Master which gives him the ability to restore it on another Jenkins machine and to have it in version control allowing rollback if there is a problem. We decided to help Rohan solve this problem.

## 1.1 Goals

After evaluating several plugin options, we started working with a plugin called JobConfigHistory. This plugin originally saves a copy of the configuration file of a job with every change made. We decided to add on to this plugin by pushing out those changes to subversion. We wanted to be able to view diffs of the configuration over time as well as allow users to load the configuration into a new Jenkins machine. In theory, this could also be used to keep two Jenkins machines in sync, allowing a user to test out a change to a config on one machine and then push it out to the others once it was successful.

## 2 Architecture

To help Rohan solve this problem, we first wanted to ensure that changes to the Job Config file were recognized as needing to be synced. In order to do this, there were two tasks that needed to be completed. Essentially what would happen is the system should automatically commit the configuration file to a Subversion repository whenever a job configuration file is modified. One of which was modifying the index.jelly to let user notice that the config file has been changed. In order to do this the index.jelly in the JobConfigHistoryProjectAction was modified to add a save button. When the user clicks the save button in the job configuration page, the onchange method in

the `JobConfigHistorySaveableListener` will be called. This method will first create a new `SVNService` instance using the information provided by user to initialize the connection to SVN. Then the method will check whether Subversion repository is synchronized. If either the local or remote copy is missing, the method will do a checkout or commit. Then finally this method will commit the configuration file just created into the repository.

However, one thing that also had to be completed concurrently in order for a user to sync the configuration file to a subversion repo, was that we needed to allow the user to specify a sync repo url. In order to do this, we added three new fields to the project config files: `scmRepoUrl`, `scmRepoUser`, `ScmRepoPass` in the file `JobConfigHistory.java`. These three fields are used for the user to specify the repository they want to sync with as well as store the credentials to make a connection to that repository. We modified the `index.jelly` file to take all three new parameters, from the online form where the user specifies Jenkins configuration. We also had to change the configure method to read in those parameters and store them. In order to interact with the newly added fields, we wrote getter and setter methods in `JobConfigHistory.java` for other methods to use.

Additionally, we also wanted a way for users to be able to view past commits of these configuration files. While we could not fully complete this user story due to the dependencies that it required, we loaded in dummy data to ensure that the function we added worked. In order to get users to be able to view past commits, we modified `JobConfigHistoryRootAction` to add a `getRevHistory` function that returns a list of `configinfo` objects. Essentially, we built on existing code by creating a `HistoryDescr` object that contained the configuration data: Author, Author ID, Change Type, and Date. Additionally, to be able to correctly fetch the data with the existing code, we had to create a `ConfigInfo` object that contained this history descriptor as well as fields for the config name, whether the config exists and if it is a job. Since we did not need the latter two for the purpose of setting up the ability for users to view past commits, we set those fields to false for now. We also modified the `index.jelly` to display this information on a timeline for the user to view.

When a user makes a change to a commit file, they may sometimes want to add a commit message or tag to it so that they remember what change

they made to the configuration file after it syncs with the svn repository. In order to allow users to tag specific commits with extra info, we first had to modify the index.jelly file in order to add a field for the user to add a commit message and tag to a configuration change. To do this, we added code to the bottom of the existing code in the index.jelly so that it displayed on the bottom of the main JobConfigHistory page. Additionally, we had to add code to JobConfigHistoryRootAction.java in order to get the jelly changes to work. Specifically, we had to write a doSubmitMessage function which validated that the plugin was successfully able to get the commit message and tag from the user. Afterwards, we added a function called doChangeCommitMessage in JobConfigHistoryRootAction.java. This function essentially just checks to make sure we have permission to make changes, gets the commit message and tags from the StaplerRequest and then applies that to the subversion repo we are syncing the configuration to. Essentially a stapler is a library that just staples application objects to URLs, which we used to get the commit message and tag from the submitted form. Once that is done, it redirects the user to a different page.

Now that we have allowed users to make changes to Jenkins config and sync it to a svn repo, we wanted to give users the ability to rollback to a specific commit. In order to roll back to specific commits, we had to modify the JobConfigHistoryRootAction index.jelly file and add a form for each config file which would, upon clicking, call a doRollback function to roll back to that config file. We had to modify the jelly file so that only config files corresponding to specific projects and not the Jenkins instance as a whole would have to option to roll back. We also had to create the doRollback function in JobConfigHistoryRootAction.java which takes a config object as an input and finds the corresponding file in our history, then updates the the project's configuration.

In order to track synchronization statuses of all of the different jobs, we decided to implement a hash map to map the names of each project that exists on the local instance of Jenkins. The map is useful for keeping the status for each job and at the same time every status will be unique. This solution is a simple way for the requirements of unique synchronization statuses to be maintained for each of the projects. In JobConfigHistoryProjectAction, there are functions that will connect the boolean status from the hash map

to a conditional for the SVN update functions that were developed in the other user stories. We then pull the values for each project's status based upon its name to retrieve the relevant decision on whether to sync or not.

Following the MVC pattern, there are two fields at the bottom of JobConfigHistoryRootAction's index page. These fields correspond to the commit message and the commit tags. When these fields have text entered into them and the submit button following the two fields is pressed, the text from the two fields is mapped to two new fields on the main plugin object. These fields are the model of this system. The controller, the getters and setters for the fields, allow the fields to be set by two strings, and allow the full commit message (message and tags) to be retrieved. When the SVN controllers attempt to make a commit, they call upon this controller logic to retrieve the full commit message.

### 3 Design

Our plugin is setup using the Model View Controller design pattern.

Our model takes the form of configuration settings on the main Jenkins plugin object, as well as new fields on the main Jenkins plugin object.

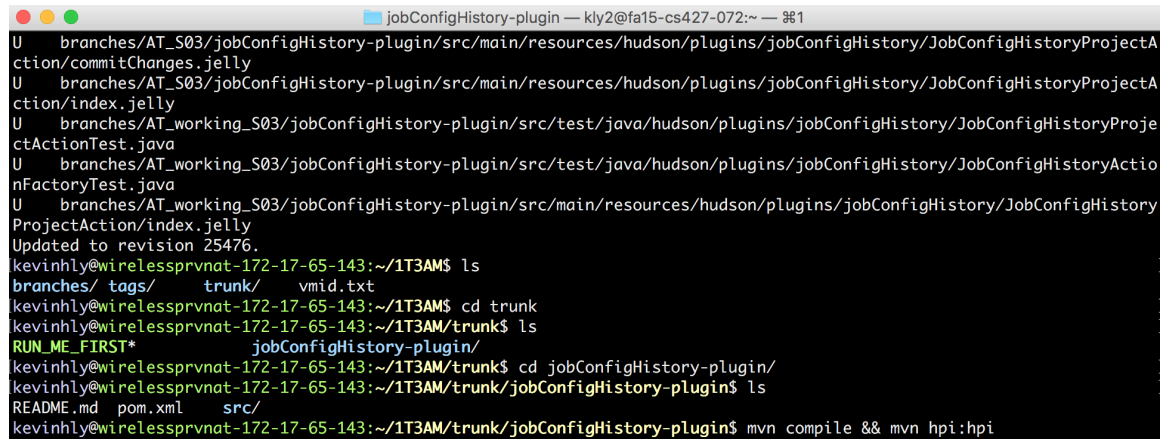
The logic of our program happens loosely in a few controllers. One controller interacts with SVN repositories: the SVNService and SVNUtil classes. These classes are called upon from another controller, the JobConfigHistorySaveableListener class. This triggers SVN related actions when Jenkins job configuration files change.

A final controller exists in the JobConfigHistoryRootAction class. This control logic reads model variables and populates values when interacting with the SVN controller classes.

Finally, our views come in three parts. The setting page of Jenkins reads from and writes to the Jenkins configuration settings. This allows you to set the repository with which to synchronize. We also allow the user to read and write two fields on the main Jenkins plugin object, for a commit message and for commit tags via a view on the main plugin page. This view is defined in

JobConfigHistoryRootActions index.jelly template.

## 4 Usage



```
jobConfigHistory-plugin — kly2@fa15-cs427-072:~ — 81
U   branches/AT_S03/jobConfigHistory-plugin/src/main/resources/hudson/plugins/jobConfigHistory/JobConfigHistoryProjectAction/commitChanges.jelly
U   branches/AT_S03/jobConfigHistory-plugin/src/main/resources/hudson/plugins/jobConfigHistory/JobConfigHistoryProjectAction/index.jelly
U   branches/AT_working_S03/jobConfigHistory-plugin/src/test/java/hudson/plugins/jobConfigHistory/JobConfigHistoryProjectActionTest.java
U   branches/AT_working_S03/jobConfigHistory-plugin/src/test/java/hudson/plugins/jobConfigHistory/JobConfigHistoryActionFactoryTest.java
U   branches/AT_working_S03/jobConfigHistory-plugin/src/main/resources/hudson/plugins/jobConfigHistory/JobConfigHistoryProjectAction/index.jelly
Updated to revision 25476.
kevinhly@wirelessprvnat-172-17-65-143:~/1T3AM$ ls
branches/ tags/      trunk/      vmid.txt
kevinhly@wirelessprvnat-172-17-65-143:~/1T3AM$ cd trunk
kevinhly@wirelessprvnat-172-17-65-143:~/1T3AM/trunk$ ls
RUN_ME_FIRST*  jobConfigHistory-plugin/
kevinhly@wirelessprvnat-172-17-65-143:~/1T3AM/trunk$ cd jobConfigHistory-plugin/
kevinhly@wirelessprvnat-172-17-65-143:~/1T3AM/trunk/jobConfigHistory-plugin$ ls
README.md  pom.xml   src/
kevinhly@wirelessprvnat-172-17-65-143:~/1T3AM/trunk/jobConfigHistory-plugin$ mvn compile && mvn hpi:hpi
```

Figure 1: Get the HPI needed to install the plugin by going into trunk/jobConfigHistory-plugin and typing "mvn compile && mvn hpi:hpi"

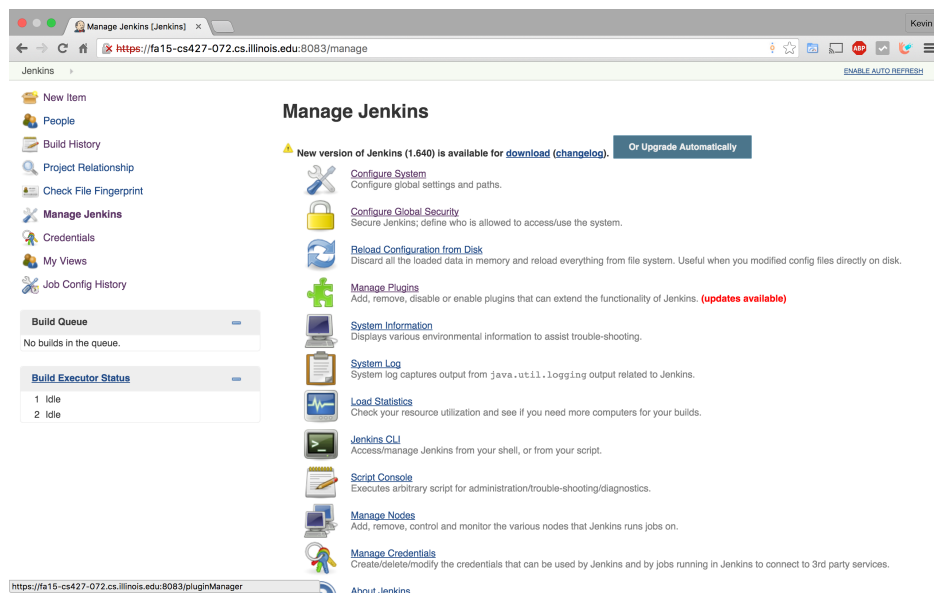


Figure 2: Go into your Jenkins instance and go to Manage Jenkins->Manage Plugins



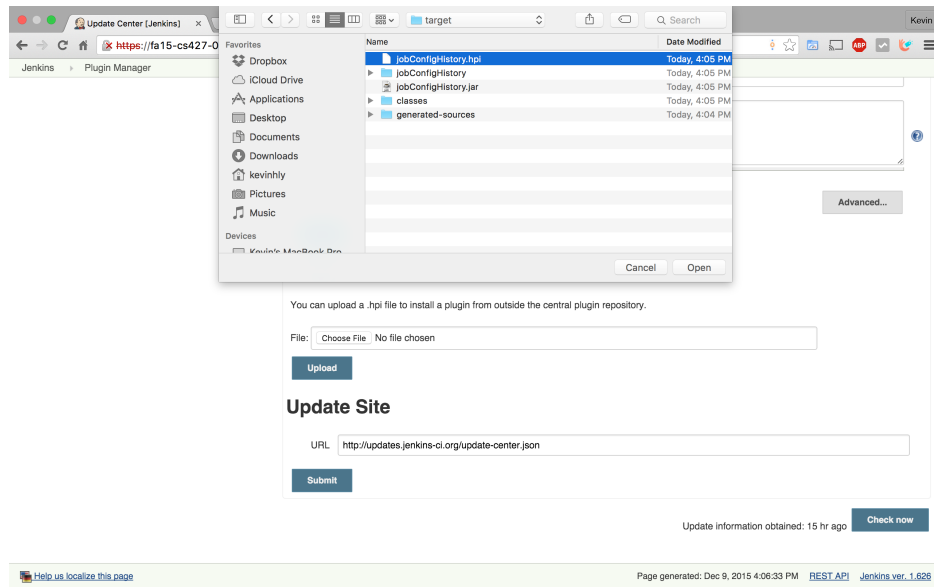


Figure 3: Get the hpi file by going into `/trunk/jobConfigHistory-plugin/target`

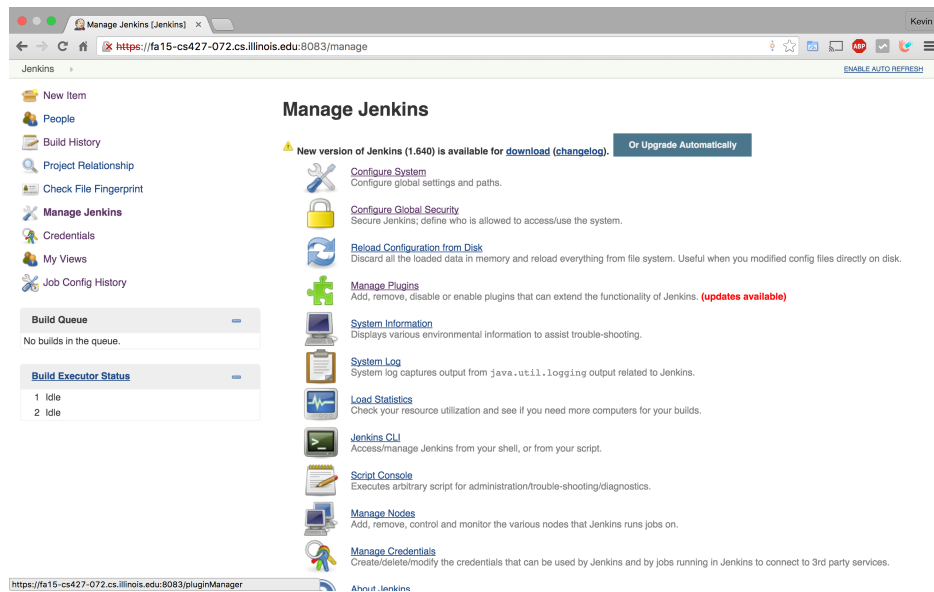


Figure 4: Go back to the Manage Jenkins page and click Configure System

Jenkins > configuration

Updates Not Installed  
Some updates could not be installed automatically.  
Install

Credentials

My Views

Job Config History

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

# of executors: 2

Labels:

Usage: Utilize this node as much as possible

Quiet period: 5

SCM checkout retry count: 0

☐ Restrict project naming

Global properties

☐ Environment variables

☐ Tool Locations

Job Config History

Use different history directory than default:

SCM Repository URL: https://subversion.assembla.com/svn/cs427test/

SCM Repository Username: cs427team

SCM Repository Password: .....

Advanced...

Maven Configuration

Default settings provider: Use default maven settings

Default global settings provider: Use default maven global settings

Save Apply

Figure 5: Enter the details of the SVN Repository You Want To Sync To

Jenkins > Job Config History

Updates Not Installed  
Some updates could not be installed automatically.  
Install

2015-12-09_16-15-14	hudson.scm.SubversionSCM (system)	Changed	kly2	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09_16-15-14	hudson.tasks.Shell (system)	Changed	kly2	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09_16-15-14	hudson.plugins.email.ExtendedEmailPublisher (system)	Changed	kly2	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09_16-15-14	hudson.tasks.Mailer (system)	Changed	kly2	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09_16-15-14	hudson.triggers.SCMTrigger (system)	Changed	kly2	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09_16-15-14	config (system)	Changed	kly2	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09_16-07-09	queue (system)	Changed	SYSTEM	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-11-17_23-44-41	queue (system)	Changed	SYSTEM	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-11-15_00-09-58	queue (system)	Changed	SYSTEM	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-10-27_12-41-45	nodeMonitors (system)	Changed	anonymous	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-10-27_12-41-42	hudson.model.UpdateCenter (system)	Changed	anonymous	<a href="#">View as XML (RAW)</a>	Cannot Rollback System Config

Commit Message

TestCommitMessage

[Tags](#)

[TestCommitTag](#)

[Submit](#)

Help us localize this page

Page generated: Dec 9, 2015 4:46:15 PM [BEST API](#) Jenkins ver. 1.626

Figure 6: Go to Your Jenkins Home Page and Click on the JobConfigHistory Tab and enter a commit message/tag

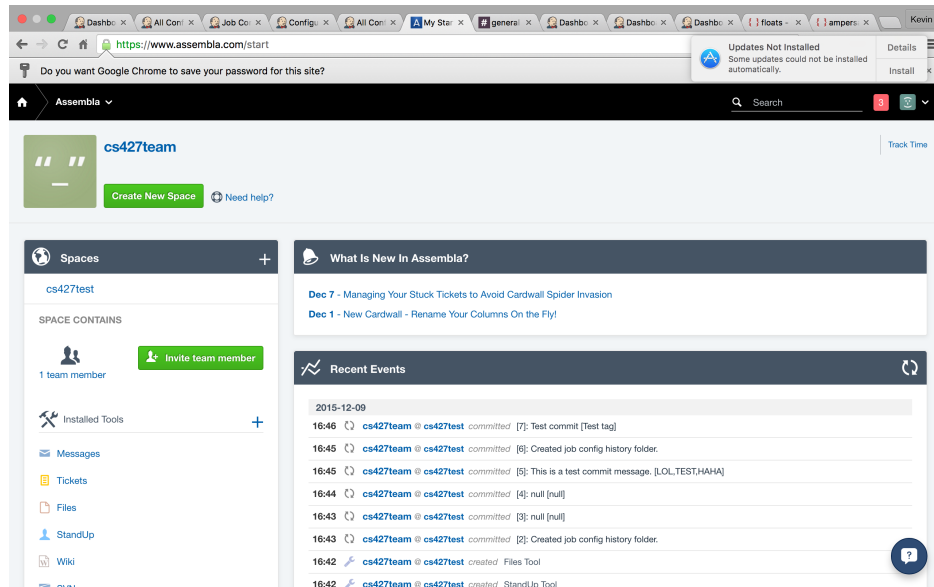


Figure 7: Go to your SVN URL and see that the commit message was saved

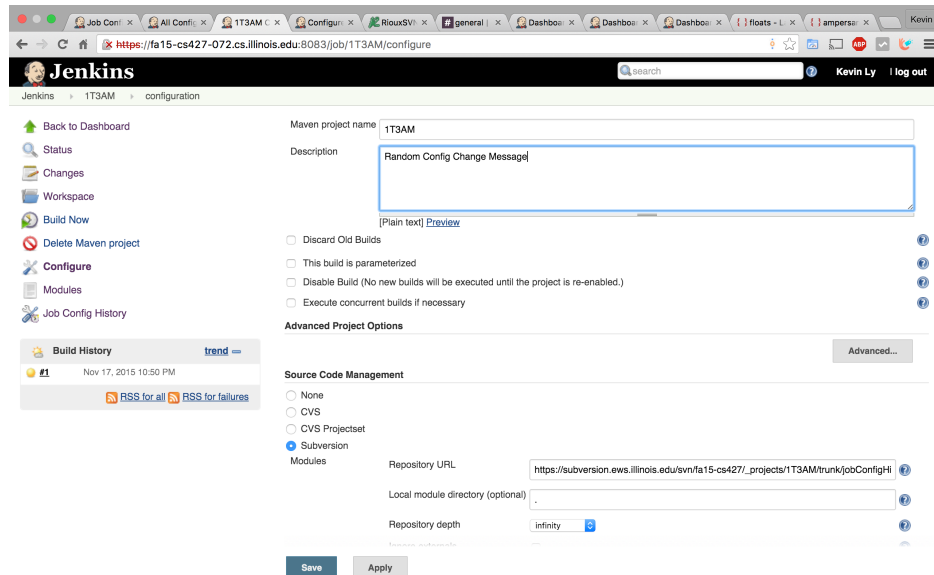


Figure 8: Go to a Project and go to the Configuration Tab and Make a Change


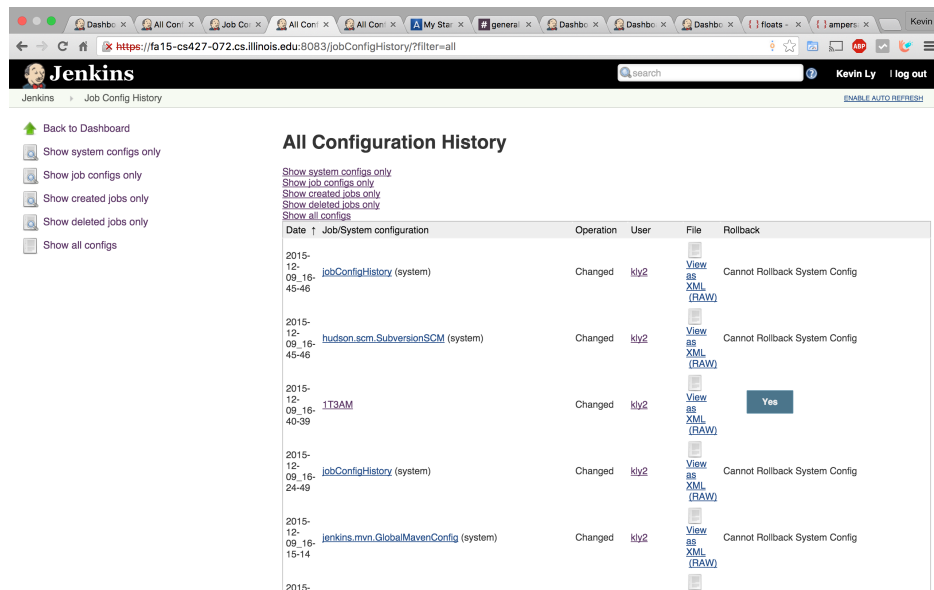
Date ↑	Operation	User	Show File	Restore old config	File A	File B
2015-12-09_16-40-39	Changed	<a href="#">kly2</a>	 <a href="#">View as XML (RAW)</a>		<input type="radio"/>	<input checked="" type="radio"/>

Figure 9: As you can see, a change was made



**All Configuration History**

[Show system configs only](#)  
[Show job configs only](#)  
[Show created jobs only](#)  
[Show deleted jobs only](#)  
[Show all configs](#)






Date ↑	Job/System configuration	Operation	User	File	Rollback
2015-12-09 16:45:46	<a href="#">jobConfigHistory</a> (system)	Changed	<a href="#">kly2</a>	 <a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09 16:45:46	<a href="#">hudson.scm.SubversionSCM</a> (system)	Changed	<a href="#">kly2</a>	 <a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09 16:40:39	<a href="#">173AM</a>	Changed	<a href="#">kly2</a>	 <a href="#">View as XML (RAW)</a>	<input checked="" type="button" value="Yes"/>
2015-12-09 16:24:49	<a href="#">jobConfigHistory</a> (system)	Changed	<a href="#">kly2</a>	 <a href="#">View as XML (RAW)</a>	Cannot Rollback System Config
2015-12-09 16:15:14	<a href="#">jenkins.mvn.GlobalMavenConfig</a> (system)	Changed	<a href="#">kly2</a>	 <a href="#">View as XML (RAW)</a>	Cannot Rollback System Config

Figure 10: Go back to the Jenkins home page and go to the Job Config History Tab->Show all configs and click on Yes for a configuration that you want to rollback to