Laboratorio de Redes y Sistemas operativos

Integración Continua en Hudson

2/12/2013

Alumnos

- **Diego Turchak**
- **Martín Cappa**

Integración Continua

Introducción

La Integración Continua es una práctica en desarrollo de software donde los miembros de un equipo integran su trabajo en forma automática tran frecuentemente como sea posible, normalmente cada persona integra al menos una vez al día llevando a múltiples integraciones por día. Cada integración es verificada por un proceso de build y testeo automático para detectar errores tan rápido como sea posible.

Requisitos

Si bien la Integración Continua es una práctica que no requiere ninguna herramienta, resulta muy útil utilizar una. Entre las más conocidas se encuentran: *Bamboo, Continuum, Hudson, Jenkins, CruiseControl* o *Anthill* o *CruiseControl.Net, Team Foundation Build* para .Net. La herramienta elegida para este trabajo fue *Hudson*.

Ventajas

- Detección de errores a través del test de unidad antes que lleguen a producción.
- Detección de código roto o incompatible.
- Detección de conflictos en los cambios.
- Disponibilidad continua de una build actual para testing, demo y puesta en producción.
- Feedback inmediato al desarrollador de la calidad, funcionalidad e impacto del código que están escribiendo.
- Frecuentes actualizaciones obligan al desarrollador a crear código modular y menos complejo.

Desventajas

- Requiere un tiempo de preparación de las herramientas antes de comenzar.
- Necesidad de desarrollar una buena suite de tests automáticos.

Procedimiento

A continuación vamos a describir el procedimiento habitual que lleva a cabo un desarrollador que participa de un proyecto implementado con una herramienta de Integración Continua.

En nuestro caso asumiremos que somos un desarrollador y que a su vez

formamos parte de un equipo de trabajo e iniciamos nuestra tarea recibiendo requerimiento de desarrollo. El primer paso que debemos hacer es copiarnos en nuestra computadora los códigos fuente de la aplicación desde el repositorio principal, esto se llama *check-out*.

Una vez que tenemos el proyecto en nuestra computadora realizamos el desarrollo requerido, luego corremos el building automatizado en la computadora de desarrollo. Si todos los tests son satisfactorios se considera una buena build. Una vez que la build está funcionando correctamente se suben los cambios al repositorio, esto se llama *commit*.

Con el repositorio actualizado usamos *Hudson* para correr el building en una máquina de integración basada en la línea principal del código. Esta build puede ser corrida por orden del desarrollador o en forma automática. Una vez que la build en el equipo de integración funciona correctamente se considera el desarrollo como finalizado.

Building automático

La Integración Continua no nos imposibilita de usar un IDE con su respectivo administrador de build. Es válido usarlas en el momento de realizar el build en la computadora de desarrollo, sin embargo es importante que el proceso de generación de la build principal se realice fuera del IDE. Para asegurar que el procedimiento pueda ejecutarse en el servidor de desarrollo.

Testeo automático

Una forma de capturar los bugs rápida y eficientemente es incluyendo tests automáticos en el proceso de build. Si bien los tests automáticos no son infalibles, filtraran una gran cantidad de bugs.

El objetivo del test es probar el build bajo condiciones controladas para detectar el problema antes que llegue a producción. Una parte significativa del test es el entorno que se usa en producción. Si se testea en un entorno diferente, cada diferencia se convierte en un riesgo que lo que se pruebe en test no se corresponda con lo que vaya a pasar en producción. Por este motivo el entorno de test debe ser tan parecido al de producción como sea posible. Se recomienda usar el mismo hardware y la misma versión de software usada en producción, principalmente base de datos, sistema operativo y toda librería utilizada por la aplicación, cómo así también las librerías que el sistema no esté usando pero se encuentren instaladas en producción.

Commit

Es el proceso por el cual el desarrollador actualiza el repositorio con los cambios que él realizó. El principal requisito que debe cumplir el desarrollador antes de hacer *commit* de su código es poder realizar una build correcta en su computadora incluyendo los tests automáticos. En todo ciclo de commit el primer paso del desarrollador debe ser actualizar su copia de trabajo para asegurarse que nadie haya subido al repositorio un cambio que sea incompatible con los que él mismo está realizando. Posteriormente vuelve a ejecutar el proceso de build, en caso de que surja algún problema debe corregirlo y volver a correr el build tantas veces como sea necesario para obtener un ejecutable funcional y que a su vez pase todos los tests. Una vez logrado ya puede realizar el commit.

Instalación Hudson

La instalación en Ubuntu Linux es muy simple, sólo requiere correr los siguientes comandos:

```
sudo sh -c "echo 'deb http://hudson-ci.org/debian /' > /etc/apt/sources.list.d/hudson.list"
sudo apt-get update
sudo apt-get install hudson
```

Los comandos para inicio y parado del servicio son:

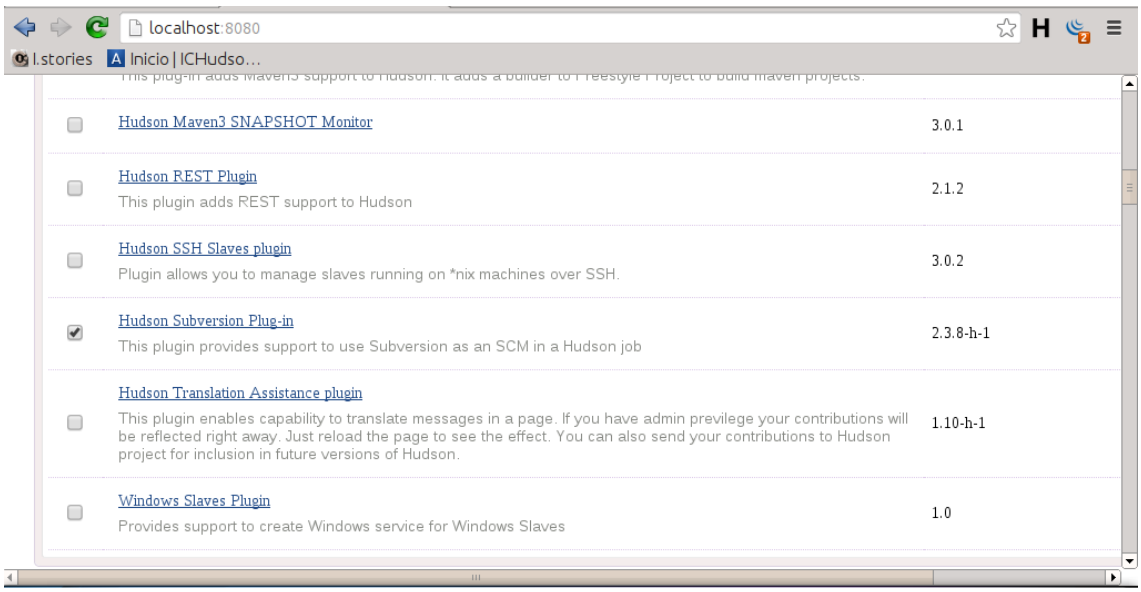
```
sudo /etc/init.d/hudson start
sudo /etc/init.d/hudson stop
sudo /etc/init.d/hudson force-reload
```

El archivo de configuración se instala en forma predeterminada en: /etc/default/hudson, transcribimos un ejemplo de configuración:

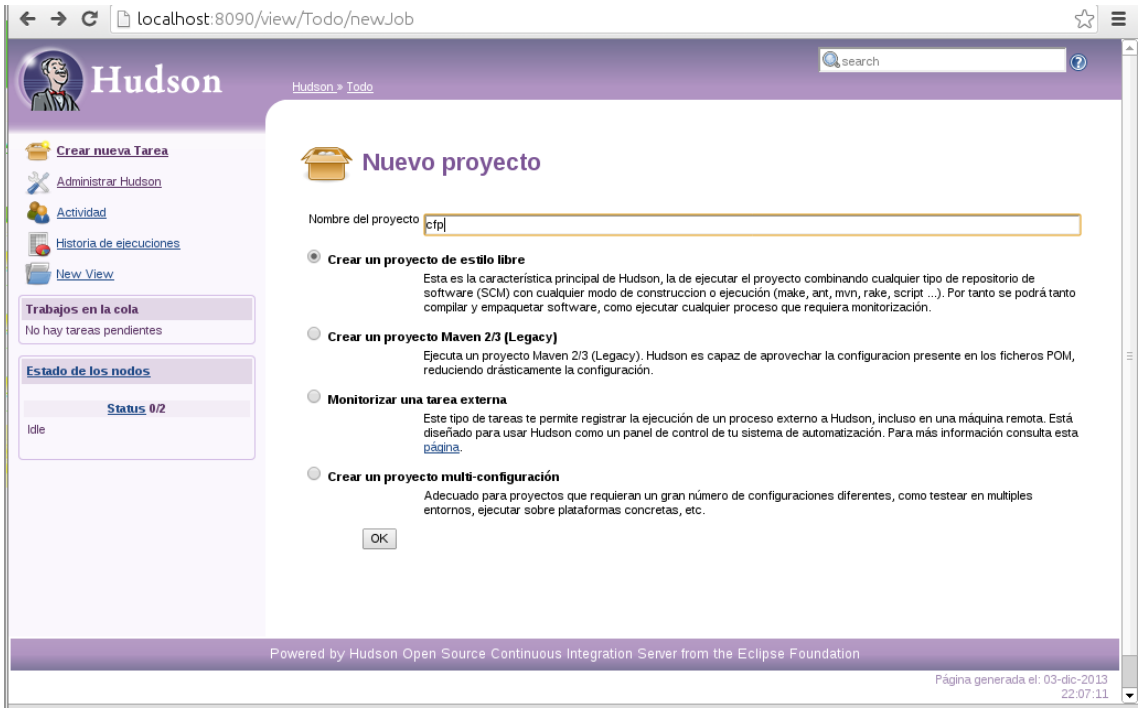
```
# defaults for hudson continuous integration server

# pulled in from the init script; makes things easier.
NAME=hudson

# location of java
JAVA=/usr/bin/java

# arguments to pass to java
#JAVA_ARGS="-Xmx256m"

PIDFILE=/var/run/hudson/hudson.pid

# user id to be invoked as (otherwise will run as root; not wise!)
HUDSON_USER=hudson

# location of the hudson war file
HUDSON_WAR=/usr/share/hudson/hudson.war

# hudson home location
HUDSON_HOME=/var/lib/hudson

# set this to false if you don't want Hudson to run by itself
# in this set up, you are expected to provide a servlet containr
# to host hudson.
RUN_STANDALONE=true

# log location.  this may be a syslog facility.priority
HUDSON_LOG=/var/log/hudson/$NAME.log
#HUDSON_LOG=daemon.info

# OS LIMITS SETUP
#   comment this out to observe /etc/security/limits.conf
#   this is on by default because
http://github.com/feniix/hudson/commit/d13c08ea8f5a3fa730ba174305e6429b74853927
#   reported that Ubuntu's PAM configuration doesn't include pam_limits.so, and as a result the
# of file
```

```
#   descriptors are forced to 1024 regardless of /etc/security/limits.conf
MAXOPENFILES=8192

# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8090

# port for AJP connector (disabled by default)
AJP_PORT=-1

# arguments to pass to hudson.
# --javahome=$JAVA_HOME
# --httpPort=$HTTP_PORT (default 8080; disable with -1)
# --httpsPort=$HTTP_PORT
# --ajp13Port=$AJP_PORT
# --argumentsRealm.passwd.$ADMIN_USER=[password]
# --argumentsRealm.$ADMIN_USER=admin
# --webroot=~/.hudson/war
HUDSON_ARGS="--webroot=/var/run/hudson/war --httpPort=$HTTP_PORT
--ajp13Port=$AJP_PORT -Djava.awt.headless=true"
```

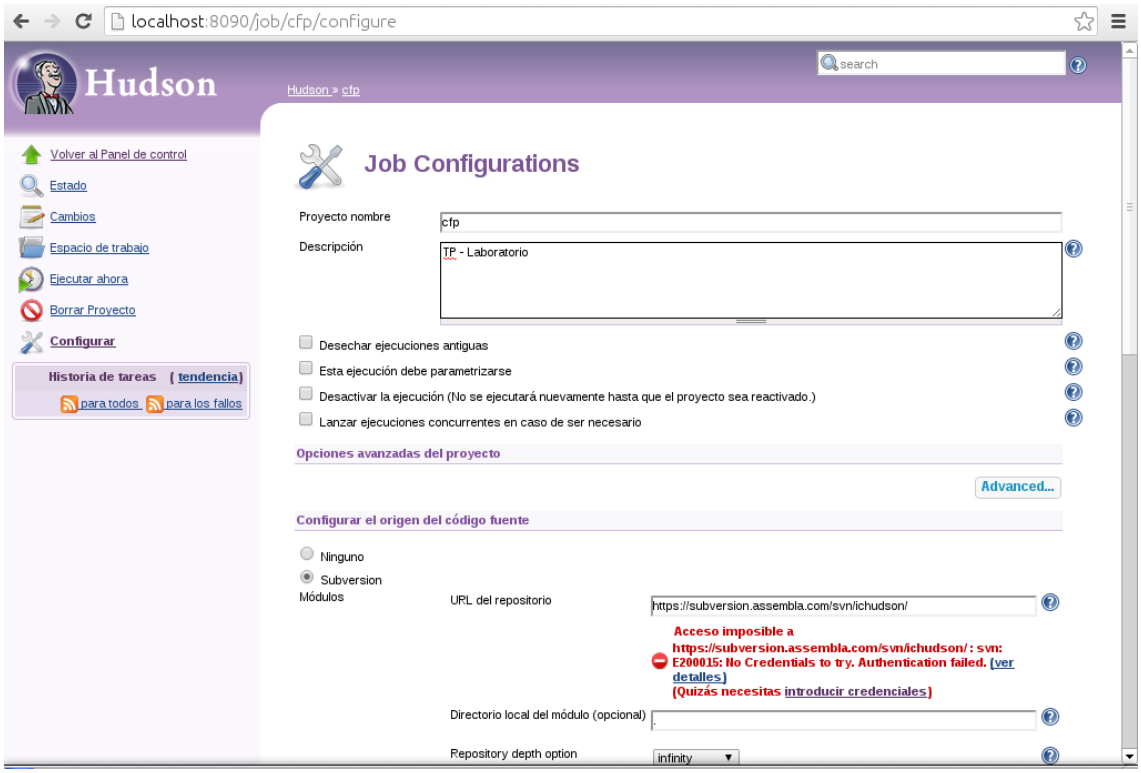
Una vez instalado accedemos a traves del navegador, la primera vez nos va a pedir que seleccionemos los plugins que vamos a usar, estos plugins se bajan e instalan automáticamente. Se muestra a continuación la selección de plugins:



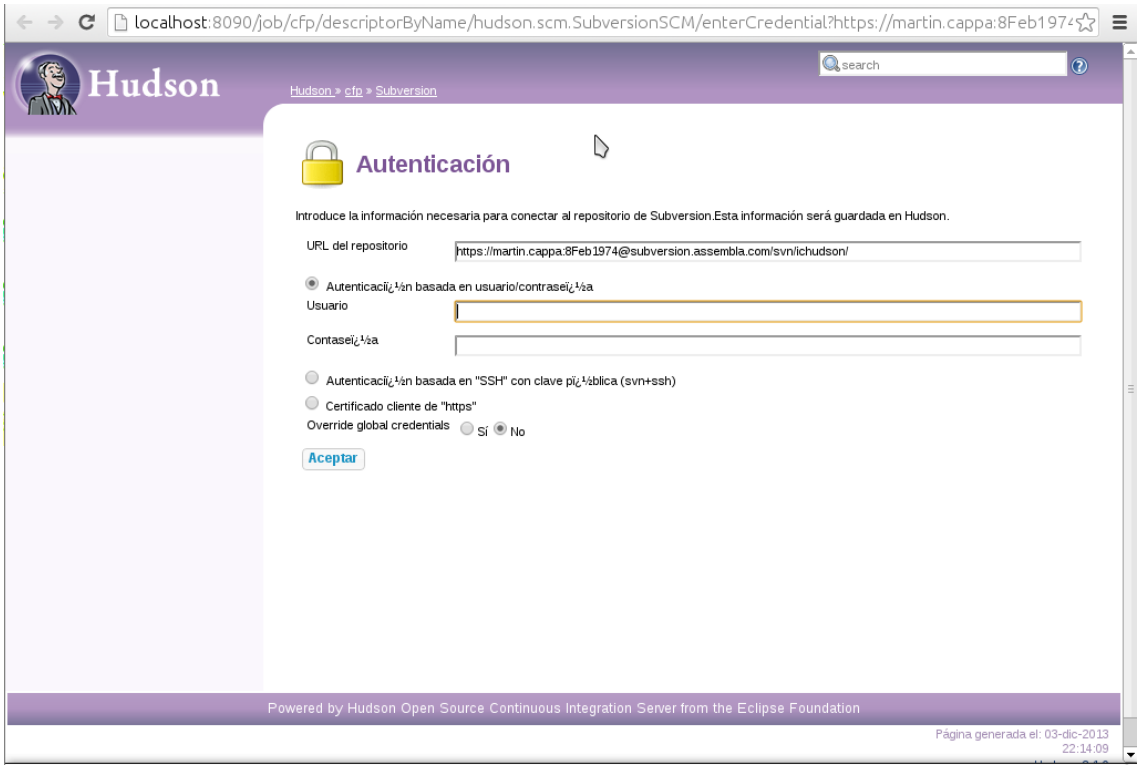
La creación del proyecto también se realiza a traves del navegador:



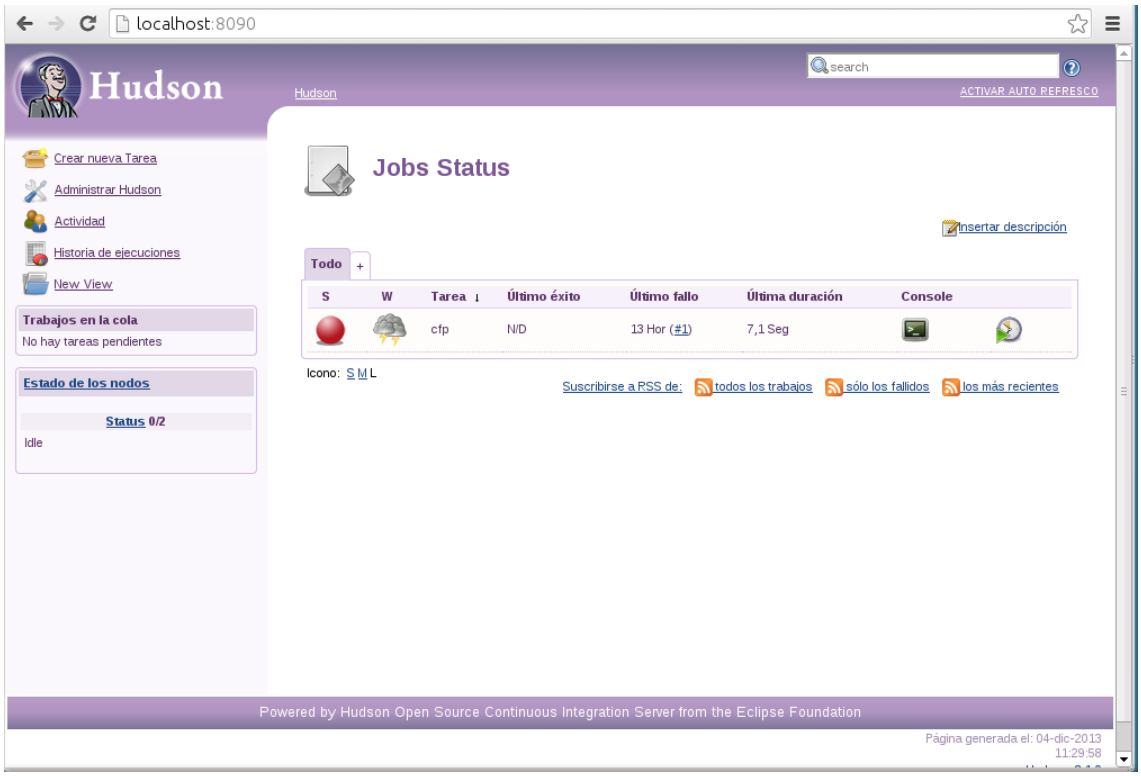
Configuración del proyecto:



La ingresar la dirección del repositorio, Hudson nos muestra un mensaje de error de acceso, esto sucede porque nuestro repositorio requiere autenticación:



Una vez configurado el proyecto se visualiza en la lista de trabajos:



Bibliografía

- http://hudson-ci.org/
- www.martinfowler.com/articles/continuousIntegration.html
- jesuslc.com/2013/04/30/ventajas-yo-puntos-fuertes-de-la-integracion-continua/
- http://en.wikipedia.org/wiki/Continuous_integration
- http://leanjavaengineering.wordpress.com/2010/08/31/grails-hudson-basics/
- http://wiki.hudson-ci.org/display/HUDSON/Grails+Plugin
- http://devnettips.blogspot.com.ar/2010/02/integracion-continua-con-hudson.html
- http://brigomp.blogspot.com.ar/2008/11/configurando-hudson-con-grails.html
- http://blogs.vitoria-gasteiz.org/ti/2010/10/08/integracion-continua-con-hudson/