

Hudson Continuous Integration Server

Stefan Saasen, stefan@coravy.com

Continuous Integration

- Software development practice
- Members of a team integrate their work frequently
- Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible
- <http://martinfowler.com/articles/continuousIntegration.html>

Continuous Integration (CI) Server



- Prevents „works on my machine“ bugs
- Starts building your project as soon as a new commit is made
- „Build“ can be anything from compiling source code to running automatic tests, static code analysis and creating packages. Often it is a combination of all of the above mentioned steps
- Sends notifications if a build fails

Choosing a CI server

- There are many CI server implementations out there. Most of them are targeting a particular language or framework
- There are a couple of Ruby/Rails based implementations:
 - `cruisecontrol.rb` (<http://cruisecontrolrb.thoughtworks.com/>)
 - `cerberus` (<http://cerberus.rubyforge.org/>)

Enter Hudson

- <http://hudson.dev.java.net>
- Two main objectives:
 - Building/testing software projects continuously
 - Monitoring executions of externally-run jobs

Enter Hudson f.

- Java based. Runs in any servlet container. Brings it's own servlet container
- Easy configuration (no XML - at least you don't have to touch it)
- RSS/E-Mail/IM integration
- Distributed builds
- Plugin support: Jabber, git, Twitter, Trac, Redmine...

Rails CI: Why Hudson?

- Hudson can be used for:
 - Java projects (has Junit/TestNG support, jar file fingerprints)
 - Rails/Ruby projects
 - Flex, Latex, C or whatever can be build with a command line script
 - As a replacement for cron/at
- Well suited for „multi language environments“

Hudson test drive

- `curl -OL http://hudson.gotdns.com/latest/hudson.war`
(or just download it using a browser)
- `java -jar hudson.war`
- open <http://localhost:8080/>
- **It's that simple**



Hudson

search ?

Hudson

[ENABLE AUTO REFRESH](#)

-  [New Job](#)
-  [Manage Hudson](#)
-  [People](#)
-  [Build History](#)

 [add description](#)

All +

| S | W | Job ↓ | Last Success | Last Failure | Last Duration |
|---|---|--------------------------------|-------------------|------------------|--|
|  |  | Test-Rails-Job | 7 days 21 hr (#3) | 8 days 0 hr (#1) | 25 sec  |

Icon: [S](#) [M](#) [L](#)

[Legend](#)  for all  for failures  for just latest builds

Build Queue

No builds in the queue.

Build Executor Status

| No. | Status |
|-----|--------|
| 1 | Idle |
| 2 | Idle |

[Hudson ver. 1.261](#)

Hudson

search ?

Hudson » Test-Rails-Job

ENABLE AUTO REFRESH

- [Back to Dashboard](#)
- [Status](#)
- [Changes](#)
- [Workspace](#)
- [Build Now](#)
- [Delete Project](#)
- [Configure](#)
- [Rcov report](#)
- [Rails stats report](#)

Project Test-Rails-Job

Example Rails project. Source:
 git://github.com/juretta/hudson-example-project-rails.git

[edit description](#)

- [Rcov report](#)
- [Rails stats report](#)
- [Workspace](#)
- [Recent Changes](#)
- [Latest Test Result \(no failures\)](#)

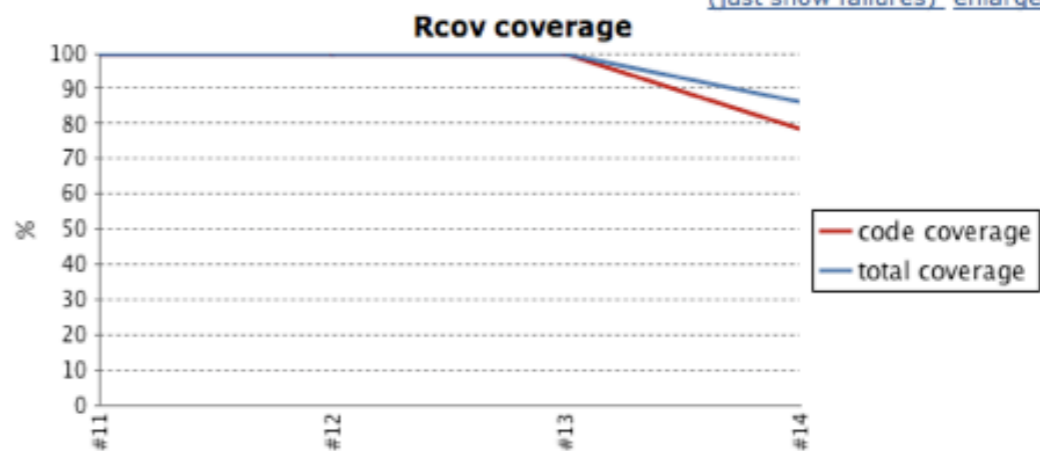
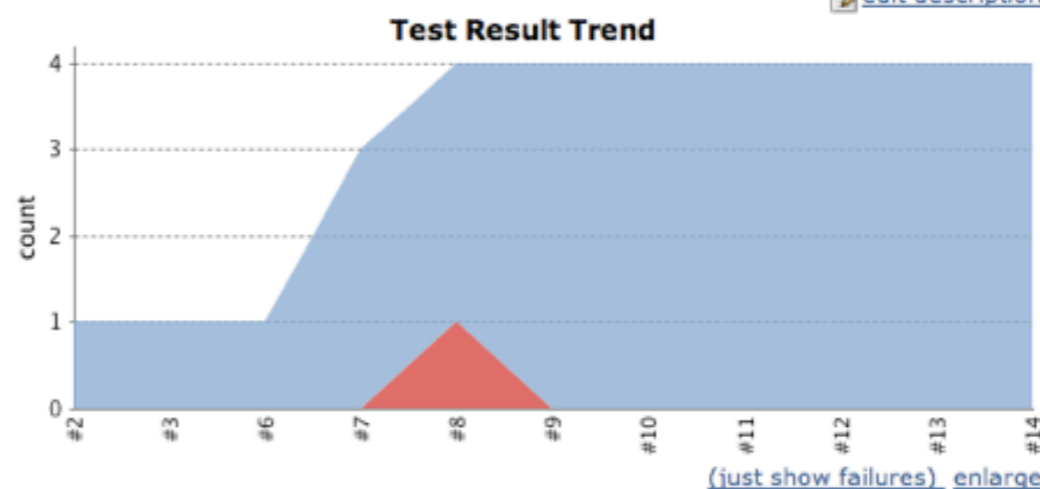
Build History [\(trend\)](#)

| | |
|-----|-------------------------|
| #14 | Nov 22, 2008 3:49:53 PM |
| #13 | Nov 22, 2008 3:47:57 PM |
| #12 | Nov 22, 2008 3:44:47 PM |
| #11 | Nov 22, 2008 3:43:47 PM |
| #10 | Nov 22, 2008 2:47:51 PM |
| #9 | Nov 22, 2008 2:47:20 PM |
| #8 | Nov 22, 2008 2:46:19 PM |
| #7 | Nov 22, 2008 2:45:12 PM |
| #6 | Nov 22, 2008 2:43:46 PM |
| #5 | Nov 22, 2008 2:43:04 PM |
| #4 | Nov 22, 2008 2:39:32 PM |
| #3 | Nov 14, 2008 5:06:57 PM |
| #2 | Nov 14, 2008 2:51:48 PM |
| #1 | Nov 14, 2008 2:17:10 PM |

[for all](#) [for failures](#)

Permalinks

- [Last build \(#14\), 4 min 52 sec ago](#)
- [Last stable build \(#14\), 4 min 52 sec ago](#)
- [Last successful build \(#14\), 4 min 52 sec ago](#)
- [Last failed build \(#8\), 1 hr 8 min ago](#)



Beim Öffnen der Seite ist ein Fehler aufgetreten. Weitere Informationen erhalten Sie im Menü „Fenster“ über die Option „Aktivität“.

How does it work?

- Configuration and projects are stored in the Hudson workspace directory
- This is `$HOME/.hudson` by default
- You can change this location simply by moving or copying the whole directory to a new location
- All the settings, logs, build archives are stored here. This is the directory to backup!
- To change this: Set the `HUDSON_HOME` environment variable before starting Hudson
`export HUDSON_HOME=/opt/local/hudson/workspace`

Rails + git + Hudson

- Install the required plugins and configure Hudson
- Prepare your Rails project
- Configure git to run Hudson builds

Install the required Hudson plugins

- To make Hudson a viable Rails build server we need a couple of plugins (Manage Hudson > Manage Plugins):
 - git
 - rake
 - ruby
 - rubyMetrics
- Install the plugins and restart Hudson!

Create a new Hudson job

The screenshot shows the Hudson web interface in a browser window. The browser's address bar shows the URL `http://localhost:8080/newJob`. The page title is "Hudson". The main content area is titled "Hudson" and contains a navigation menu on the left with links for "New Job", "Manage Hudson", "People", and "Build History". The "New Job" link is highlighted. The main form area has a "Job name" field containing "Rails-Project-Test-Job". Below this, there are five radio button options for job types: "Build a free-style software project" (selected), "Build a maven2 project", "Build multi-configuration project (alpha)", "Monitor an external job", and "Copy existing job". Each option has a brief description. At the bottom of the form is an "OK" button. On the left side, there are two summary boxes: "Build Queue" showing "No builds in the queue." and "Build Executor Status" showing a table with two executors in an "Idle" state. The footer of the page indicates "Hudson ver. 1.261".

Hudson

Job name

Build a free-style software project
This is the central feature of Hudson. Hudson will build your project, You can combine any SCM with any build system, and this can be even used for something other than software build.

Build a maven2 project
Build a maven2 project. Hudson takes advantage of your POM files and drastically reduces the configuration.

Build multi-configuration project (alpha)
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Monitor an external job
This type of job allows you to record the execution of a process run outside Hudson, even on a remote machine. This is designed so that you can use Hudson as a dashboard of your existing automation system. See [the documentation for more details.](#)

Copy existing job
Copy from

OK

Build Queue
No builds in the queue.

| No. | Status |
|-----|--------|
| 1 | Idle |
| 2 | Idle |

Hudson ver. 1.261

Configure the project

The screenshot shows the Hudson web interface for configuring a project named "Test-Rails-Job". The browser address bar shows the URL `http://localhost:8080/job/Test-Rails-Job/configure`. The interface is divided into several sections:

- Build History:** A list of 14 builds, with the most recent (#14) on Nov 22, 2008 at 3:49:53 PM. Builds #1-7 are marked as successful (blue), while #8-14 are marked as failed (red).
- Source Code Management:** The "Git" option is selected. The "Repository URL" is `file:///ruby/rails/rails-example-hudson/.git` and the "Branch" is `origin/master`. The "Repository browser" is set to "(Auto)".
- Build Triggers:** Three options are listed: "Build after other projects are built", "Poll SCM", and "Build periodically", all of which are currently unchecked.
- Build:** The "Invoke Rake" option is selected. The "Rake Version" is set to "(Default)". The "Tasks" field contains the following text:

```
ci:setup:testunit
test
test:test:rcov
CI_REPORTS=results
RAILS_ENV=test
```

At the bottom of the "Build" section, there is a button labeled "Advanced...".

Post-build Actions

- Archive the artifacts
- Record fingerprints of files to track usage
- Publish Javadoc
- Publish JUnit test result report

You need the `ci_reporter` gem for this.

Test report XMLs

results/*.xml

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is [the workspace root](#).

- Aggregate downstream test results
- Build other projects
- E-mail Notification

Recipients

stefan@coravy.com

Whitespace-separated list of recipient addresses. E-mail will be sent when a build fails.

- Send e-mail for every unstable build
- Send separate e-mails to individuals who broke the build

- Push GIT tags back to origin repository
- Publish Rcov report

Rcov report directory

doc/coverage/test

relative path to the coverage report directory

Coverage metric targets

| | | | | | |
|-------------------------|----|---|---|---|---|
| Total coverage ☀ | 80 | ☁ | 0 | 🟡 | 0 |
| Code coverage ☀ | 80 | ☁ | 0 | 🟡 | 0 |

Configure health reporting thresholds.
For the ☀ row, leave blank to use the default value (i.e. 80).
For the ☁ and 🟡 rows, leave blank to use the default values (i.e. 0).

- Publish Rails stats report

Rake Version

(Default)

Save

Rails + git + Hudson

- Install the required plugins and configure Hudson
- **Prepare your Rails project**
- Configure git to run Hudson builds

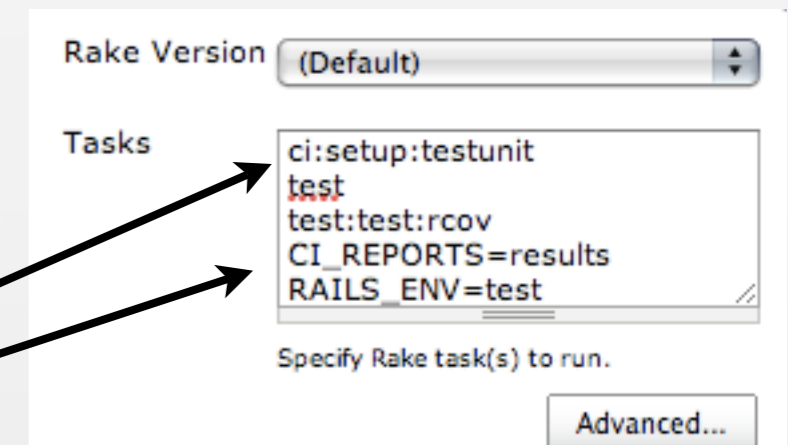
Prepare the Rails project

- There is no need to change anything if you don't want to
- Hudson can run any script and marks a build as success or failure depending on the exit code of the script (echo \$? -> 0: success, everything else: failure)
- But: to draw fancy graphs we need a machine readable test output

Prepare the Rails project f.

- „Ci::Reporter is an add-on to Test::Unit and RSpec that allows you to generate XML reports of your test and/or spec runs. The resulting files can be read by a continuous integration system that understands Ant's JUnit report XML format.“ (http://caldersphere.rubyforge.org/ci_reporter/)
- Install the ci_reporter gem: `sudo gem install ci_reporter`
- Add „ci_reporter“ to the main Rakefile and invoke the ci_reporter target

```
11 ~
12 require 'rubygems'~
13 gem 'ci_reporter'~
14 require 'ci/reporter/rake/test_unit' # use this if you're using Test::Unit~
15
```



Rake Version (Default)

Tasks

- ci:setup:testunit
- test
- test:test:rcov
- CI_REPORTS=results
- RAILS_ENV=test

Specify Rake task(s) to run.

Advanced...

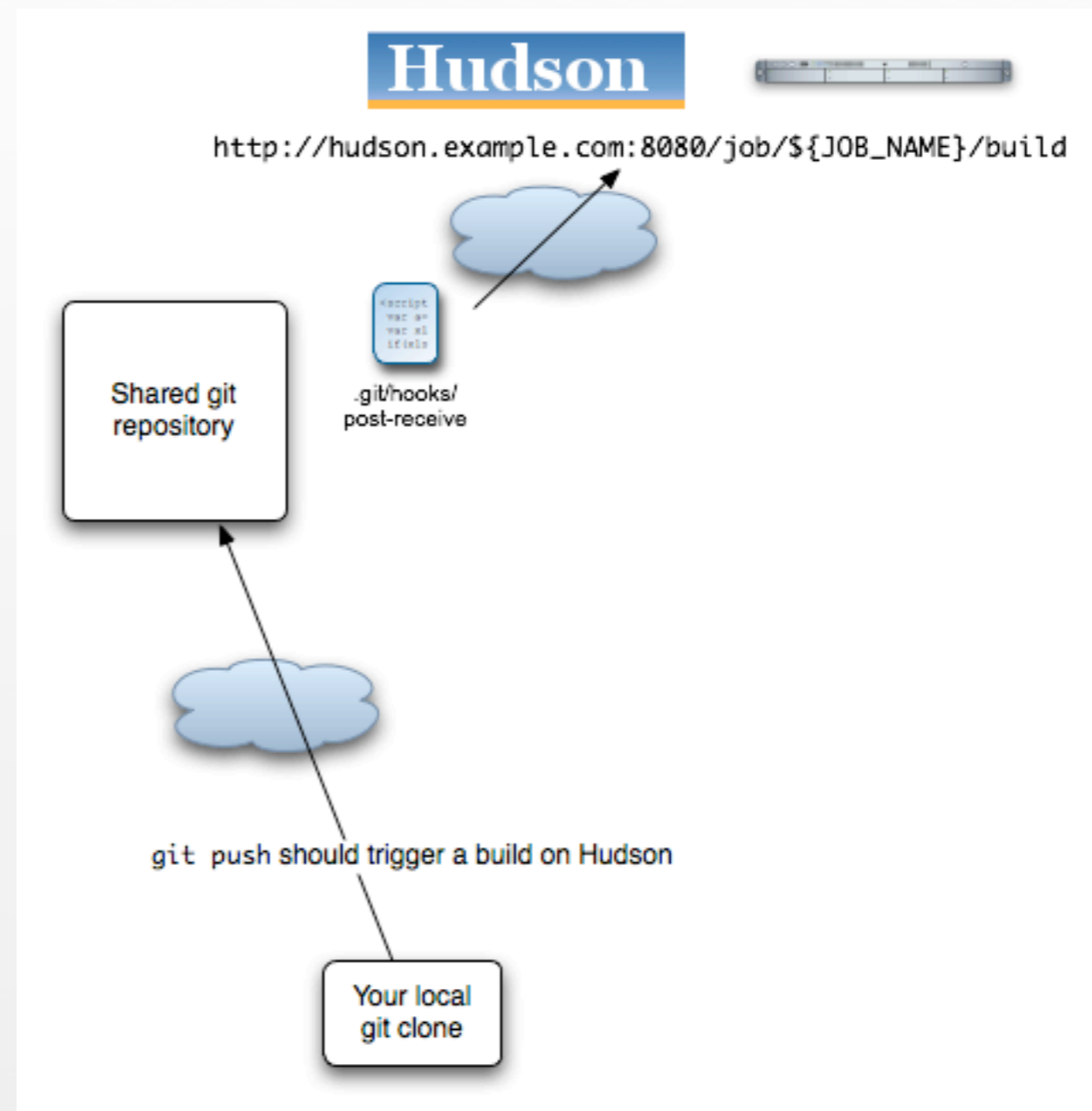
ci_reporter

Rails + git + Hudson

- Install the required plugins and configure Hudson
- Prepare your Rails project
- **Configure git to run Hudson builds**

Configure git to run Hudson builds

- Run a build as soon as you have pushed your changes back to the shared repository



Configure git to run Hudson builds f.

- Add the following lines to the `.git/hooks/post-receive` hook (Change the job name accordingly):

```
5 URL='http://localhost:8080/job/Test-Rails-Job/build'-  
6 echo "Run Hudson build: ${URL}"-  
7 curl| $URL > /dev/null 2>&1-  
8 -
```

- Make the post-receive hook executable:
`chmod u+x .git/hooks/post-receive`

Thank you.
Questions?