

Deploy strategies

#pugMI - 11/04/2013

Casi

- Caso 1 - Contactlab
- Caso 2
- Caso 3 - S'nce Group
- Caso 4 - Terravision
- Caso 5 - OneBip

Caso I - Scenario

- Tipologia di progetto
- Realtà operativa

Caso I - Parametri di scelta

- No Ruby o altri linguaggi
- Sistema e configurazioni embeddate nel progetto
- Semplicità | Anche per frontend developer
- Continuità con sistema precedente (Symfony 1.4)
- Non infastidire sistemi!

Caso I - Processo

1. Backup DB esistente - In base al tipo di modifiche
2. Messa offline del sito
3. Posizionamento file offline.html
4. Deploy del nuovo codice
5. doctrine:migration:migrate
NB: Possibilità di modifica e inserimento dati
6. doctrine:fixtures:load

Caso I - DeployBundle

hpatolo / DeployBundle
forked from dator/DeployBundle

Watch Star 11 Fork 6

Code Network Pull Requests 0 Issues 0 Graphs

Porting of Symfony 1.4 project:deploy command to Symfony 2 — [Read more](#)

Clone in Mac ZIP HTTP SSH Git Read-Only git://github.com/hpatolo/DeployBundle.git Read-Only access

branch: master Files Commits Branches 1 Tags 4

DeployBundle / + 28 commits

Reference to --force-vendor

hpatolo authored 2 days ago latest commit d6b24557e9

Command	2 days ago	Added post_deploy_operations and force-vendor feature [hpatolo]
DependencyInjection	2 days ago	Added post_deploy_operations and force-vendor feature [hpatolo]
Resources	2 days ago	Reference to --force-vendor [hpatolo]
DeployBundle.php	10 months ago	Added License and copyright info and applied coding standards using h... [hpatolo]
README.md	6 months ago	Updated link to the master documentation [hpatolo]
composer.json	6 months ago	Cleaning ad reorganizing [hpatolo]

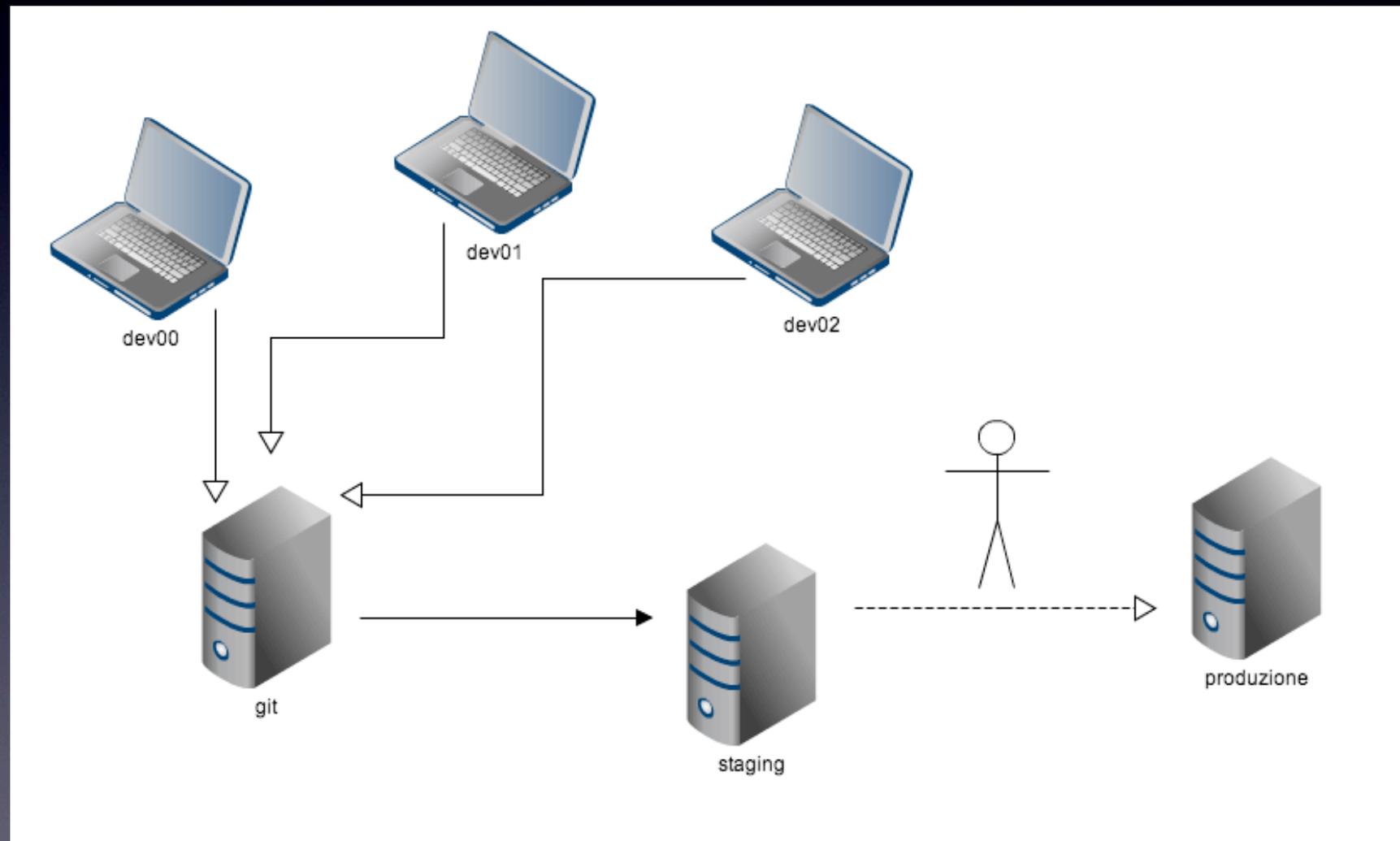
<https://github.com/hpatolo/DeployBundle>

Caso I - DeployBundle

Configurazione

```
deploy:
  server1:
    rsync-options: -azC --force --delete --progress --checksum
    host: 12.34.56.78
    user: user_ssh_macchina_remota
    dir: /path/to/prj/root/httpdocs/
    port: 22022
    post_deploy_operations:
      - app/console cache:clear --env=prod
      - app/console assets:install --env=prod
      - app/console assetic:dump --env=prod
      - rm web/offline.html
```

Caso 2 - Scenario



Caso 2 - Processo

- Tutti gli sviluppatori pushano sullo stesso branch di un repository git, che in automatico lancia un hook che sincronizza tramite **rsync** sul server di staging.
- Una volta testato (manualmente) il software viene lanciato uno script bash per sincronizzare le directory ed eseguire alcuni banali task.
- I database vengono aggiornati manualmente

Caso 2 - Esempio Rsync

```
rsync -avz --exclude 'app/config/parameters.yml' --exclude 'app/cache' --  
exclude 'app/cache_sessions' --exclude 'app/logs' . root@$remote_server:/  
var/www/devhost/htdocs
```

```
ssh root@$remote_server rm -fr /var/www/devhost/htdocs/app/cache/*  
#ssh root@$remote_server rm -fr /var/www/devhost/htdocs/app/logs/*
```

```
ssh root@$remote_server php /var/www/devhost/htdocs/app/console  
assets:install /var/www/devhost/htdocs/web --symlink
```

```
ssh root@$remote_server php /var/www/devhost/htdocs/app/console  
cache:clear
```

```
ssh root@$remote_server php /var/www/devhost/htdocs/app/console  
cache:clear --env=prod --no-debug
```

```
ssh root@$remote_server chmod -R 777 /var/www/devhost/htdocs/app/cache
```

```
ssh root@$remote_server chmod -R 777 /var/www/devhost/htdocs/app/logs
```

```
ssh root@$remote_server chmod -R 777 /var/www/devhost/htdocs/app/  
cache_sessions
```

```
ssh root@$remote_server chown -R www-data:www-data /var/www/devhost/htdocs
```

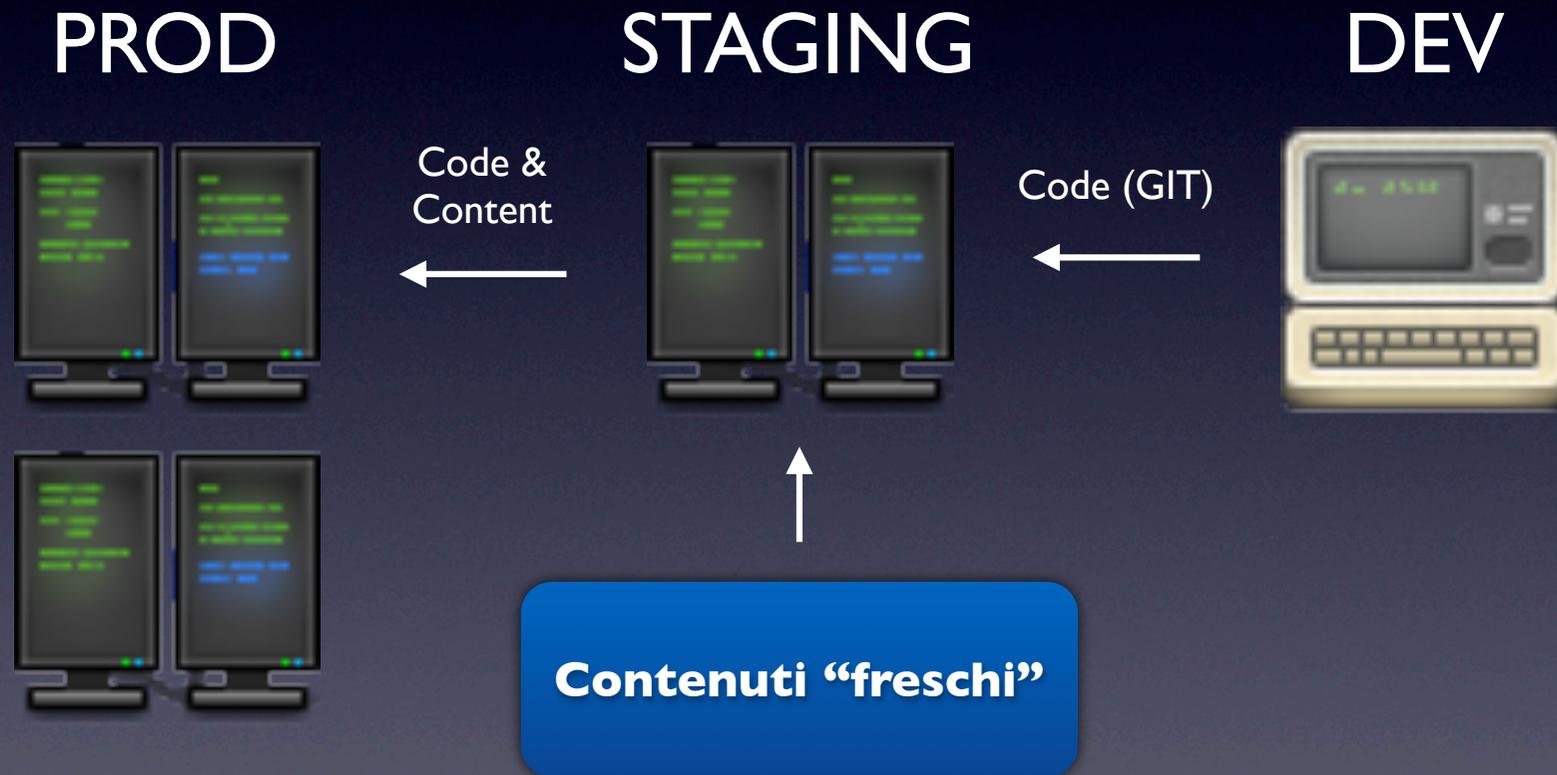
Caso 2 - Conclusioni

- **Pro:** facilmente customizzabile, poche cose da mantenere e gestire, applicabile rapidamente su tutti i server.
- **Contro:** gestione delle migrazioni del database inesistente

Caso 3 - Scenario

- Deploy di “**contenuti**”
- Ambiente di Staging è quello di riferimento
- Automatizzare il più possibile tutti i processi (non sempre possibile)

Caso 3 - Environment



Caso 3 - Deploy

- “Deploy” di nuovi contenuti attraverso estensione (Sviluppata da Gaetano) - Operazione giornaliera
- Deploy di nuovi country (siti) - **Phing**

Caso 3 - Processo

- Processo Phing:
 - Preparazione Dump MySQL e FileSystem folder
 - Copia sui server di Prod
 - Backup DB, FileSystem, SolrIndex
 - Offline
 - Sostituzione DB, FileSystem
 - Pull GIT (Server GIT interno)
 - Cache warming
 - Update SolrIndex e Niceurls
 - Test -> Pingdom (le API non supportano ancora i test transazionali)
 - **Live!**

Caso 3 - Pingdom

Test from USA
 Europe

Revisions

Check editor

1	Go to URL <code>http://www.whirlpool.se</code>	528 ms	×
2	Fill in field <code>#SearchText</code> with <code>Ugnar</code>	2 ms	×
3	Submit form <code>#CatalogSearchForm</code> ↗	2,733 ms	×
4	Wait for element <code>.product-row</code> to exist	2 ms	×

Run Test

Caso 4 - Scenario

- Terravision
- Applicazione Symfony 2
- Server Redis

- Capistrano utilizzato per il deploy
(Capistrano PHP)

Caso 4 - Processo

- Aggiornamento Staging:
 - Da locale **cap deploy**
 - Nuova release in un array definito di server
 - Pull da repository GIT
 - Copia vendor e aggiornamento
 - Asset nella CDN
 - Aggiornamento cache
 - Eventuali migrazioni e pulizia cache Redis (vengono lasciate solo le ultime 3 release)

Caso 4 - Nodi

- Chef crea un server con:
 - Nginx
 - Fastpfm
 - Postfix
 - APC
 - Parameter (x le password dell applicazione)
 - Virtual Host
 - XHProf
- Aggiunta all'array del Load Balancer