

d4r⁷

Gianni Bombelli
Mini IAD Vimercate
19 maggio 2018

INTR3

Docker alla settima

Ambiente di Continuous Delivery che gira in container docker, produce container docker sfruttando altri container docker, testa container docker usando altri container docker e rilascia container docker in container docker.

Il Team

Scrum Team composto da:

- 4 “full-stack” Software Engineer
- 1 UX e UI Designer
- 1 UI Designer e front-end Developer
- (1 Tester)

Le Applicazioni

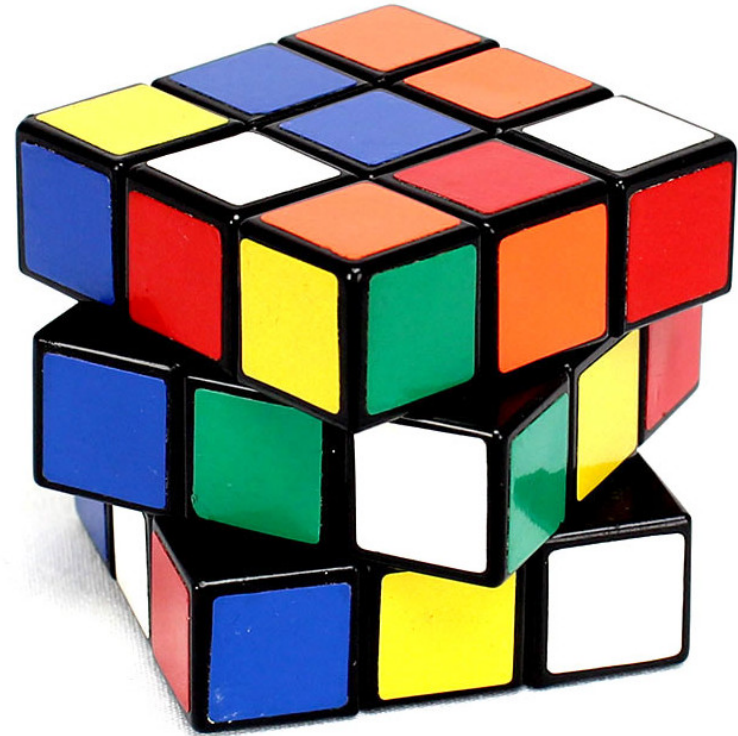
Suite composta da 8 applicazioni Web che fanno parte di un sistema ancora più vasto.

Installazioni in oltre 30 paesi sparsi per il mondo.



Supporto continuo al Team di Operation...

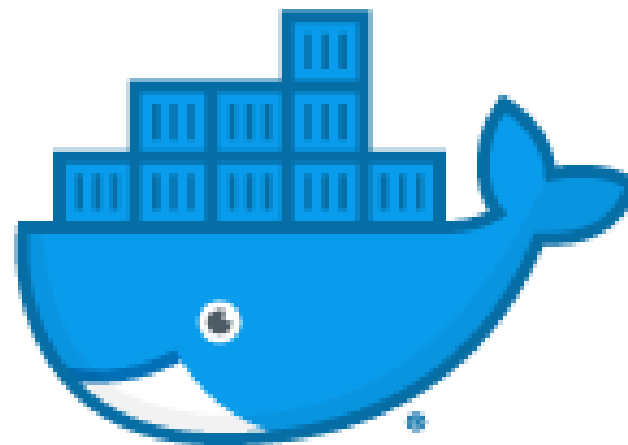
- Ogni server si è evoluto seguendo la propria via
- Problemi durante l'installazione delle dipendenze
- Errori di configurazione delle applicazioni
- Uso di configurazioni vecchie "deprecate"





Supporto continuo al Team di Operation...

- Ogni server si è evoluto seguendo la propria via
- Problemi durante l'installazione delle dipendenze
- Errori di configurazione delle applicazioni
- Uso di configurazioni vecchie “deprecate”





```
image@image:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
9bb5a5d4561a: Pull complete
Digest: sha256:f5233545e43561214ca4891fd1157e1c3c563316ed8e237750d59bde73361e77
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

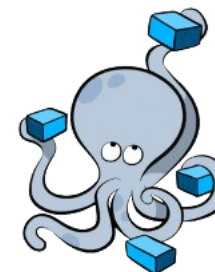
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/engine/userguide/
```



Giochiamo, sperimentiamo, ma
ci serve un obiettivo concreto...

Replichiamo il nostro build
server usando docker





10+ progetti
differenti tecnologie
diversi environment per le build





10+ progetti
differenti tecnologie
diversi environment per le build

Jenkins tool providers

Jenkins slaves

container docker come slave





Abbiamo creato 2 immagini:

- jenkins-slave
- jenkins-sencha-slave

NodeJS

NameNodeJS-6

☒ Install automatically

Install from nodejs.org

VersionNodeJS 6.14.2

Global npm packages to install

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax 'packageName@version'

Global npm packages refresh hours72

Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

Delete Installer

Add Installer

NodeJS

NameNodeJS-8

☒ Install automatically

Install from nodejs.org

VersionNodeJS 8.11.1

Global npm packages to install

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax 'packageName@version'

Global npm packages refresh hours72

Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

Delete Installer

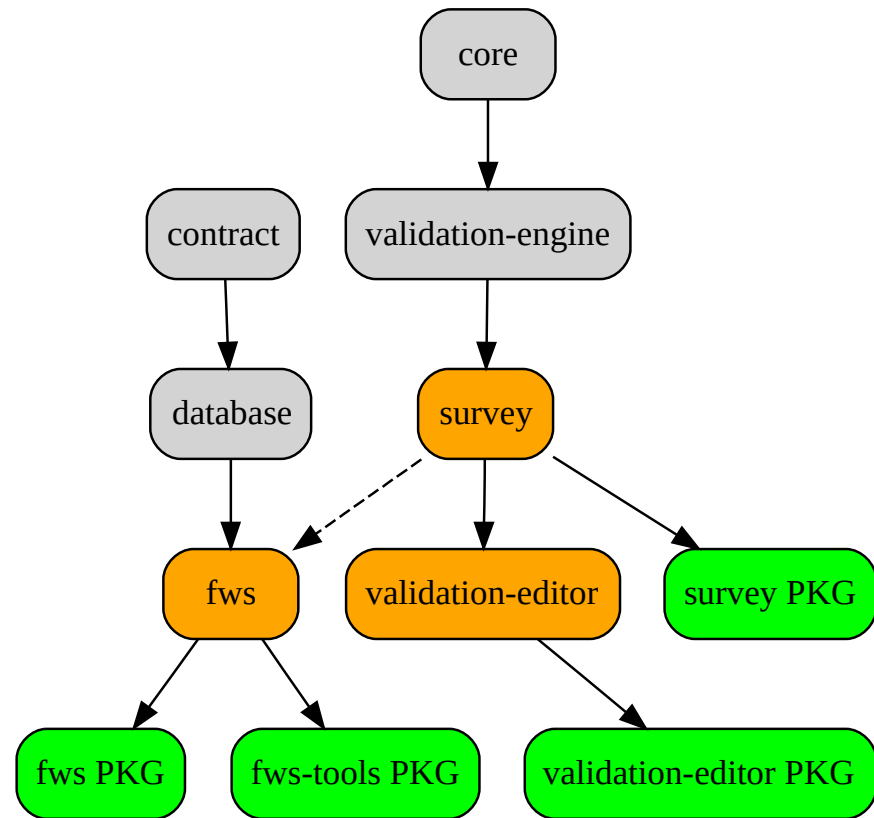
Add Installer

Add NodeJS

List of NodeJS installations on this system



dipendenze tra
applicazioni e repository





Per generare il package ci
sono tanti job in cascata

Non abbiamo un colpo
d'occhio sullo stato delle
build...

S	W	Name	Last Success	Last Failure	Last Duration
Success	Sun	build-001	5 mo 27 days - #96	9 mo 18 days - #49	1 min 32 sec
Success	Sun	build-002	5 mo 27 days - #96	9 mo 13 days - #54	44 sec
Success	Sun	build-003	1 yr 2 mo - #1	N/A	53 sec
Success	Sun	build-004	1 yr 1 mo - #3	1 yr 2 mo - #1	50 sec
Success	Sun	build-005	2 days 3 hr - #177	1 yr 5 mo - #16	2 min 0 sec
Success	Sun	build-006	1 yr 5 mo - #9	N/A	8.3 sec
Success	Sun	build-007	3 mo 11 days - #13	N/A	58 sec
Success	Sun	build-008	3 mo 5 days - #19	3 mo 22 days - #17	8 min 50 sec
Success	Sun	build-009	2 days 3 hr - #178	2 mo 19 days - #170	24 min
Success	Sun	build-010	1 yr 5 mo - #12	1 yr 6 mo - #7	20 min
Success	Sun	build-011	3 mo 11 days - #13	N/A	2 min 8 sec
Success	Sun	build-012	2 days 3 hr - #179	1 yr 6 mo - #1	1 min 49 sec
Success	Sun	build-013	1 yr 5 mo - #10	1 yr 6 mo - #3	6.8 sec
Success	Sun	build-014	3 mo 11 days - #13	N/A	32 sec
Success	Sun	build-015	3 mo 11 days - #12	3 mo 12 days - #11	8 min 58 sec
Success	Sun	build-016	6 hr 20 min - #217	3 mo 12 days - #149	45 sec
Success	Sun	build-017	1 yr 4 mo - #42	1 yr 6 mo - #4	6.1 sec
Success	Sun	build-018	6 hr 19 min - #211	5 mo 18 days - #100	51 sec
Success	Sun	build-019	1 yr 4 mo - #40	1 yr 6 mo - #5	5.7 sec
Success	Sun	build-020	6 hr 18 min - #212	4 days 1 hr - #209	13 min



dipendenze tra applicazioni
e repository:

~~job in cascata~~

build pipeline => solo 1 job
per ogni applicazione

Average stage times:
(Average full run time: ~22min 12s)

	Checkout Contract	Compile Contract	Checkout Database	Compile Database	Checkout FrontWork Space	Compile FrontWork Space	Build FrontWork Space Docker Image
#24 Apr 09 11:25 No Changes	1min 21s	8s	1min 26s	5s	2min 55s	13min 56s	1min 34s
#23 Apr 09 10:41 No Changes	49s	7s	1min 4s	5s	2min 8s	14min 10s	1min 34s
#22 Apr 09 09:44 1 commits	46s	6s	59s	5s	1min 41s	13min 47s	1min 34s
#21 Apr 06 15:56 2 commits	1min 13s	8s	53s	4s	3min 30s	14min 4s	1min 40s
#20 Apr 05 10:41 1 commits	1min 10s	8s	1min 2s	6s	3min 40s	15min 18s	1min 49s
#19 Apr 05 10:26 No Changes	41s	5s	43s	3s	1min 28s	14min 18s	1min 17s
#18 Apr 05 09:19 No Changes	50s	7s	36s	4s	1min 50s	10min 18s failed	
#17 Apr 04 17:46 No Changes	1min 40s	7s	50s	5s	2min 9s	14min 3s	1min 25s
#16 Apr 04 16:17 No Changes	2min 34s	6s	3min 6s	4s	3min 44s failed		
	2min 36s	8s	3min 16s	6s	5min 0s	14min 34s	1min 41s



Operation ha un “Agreement”
con Security Team:

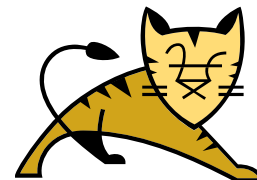
- Tutte le VM e le immagini docker devono essere basate su CentOS 7.4
- Lentezza nella build causata dalla creazione delle immagini partendo da CentOS 7.4

Passi per creare un immagine:

- Download immagine base
- Installazione dipendenze
- Configurazione ambiente interno all'immagine
- “Deploy” nell'immagine della nostra applicazione



Ottimizziamo il processo
creando delle immagini “base”
che rilasciamo sull’artifactory
interno e riutilizziamo

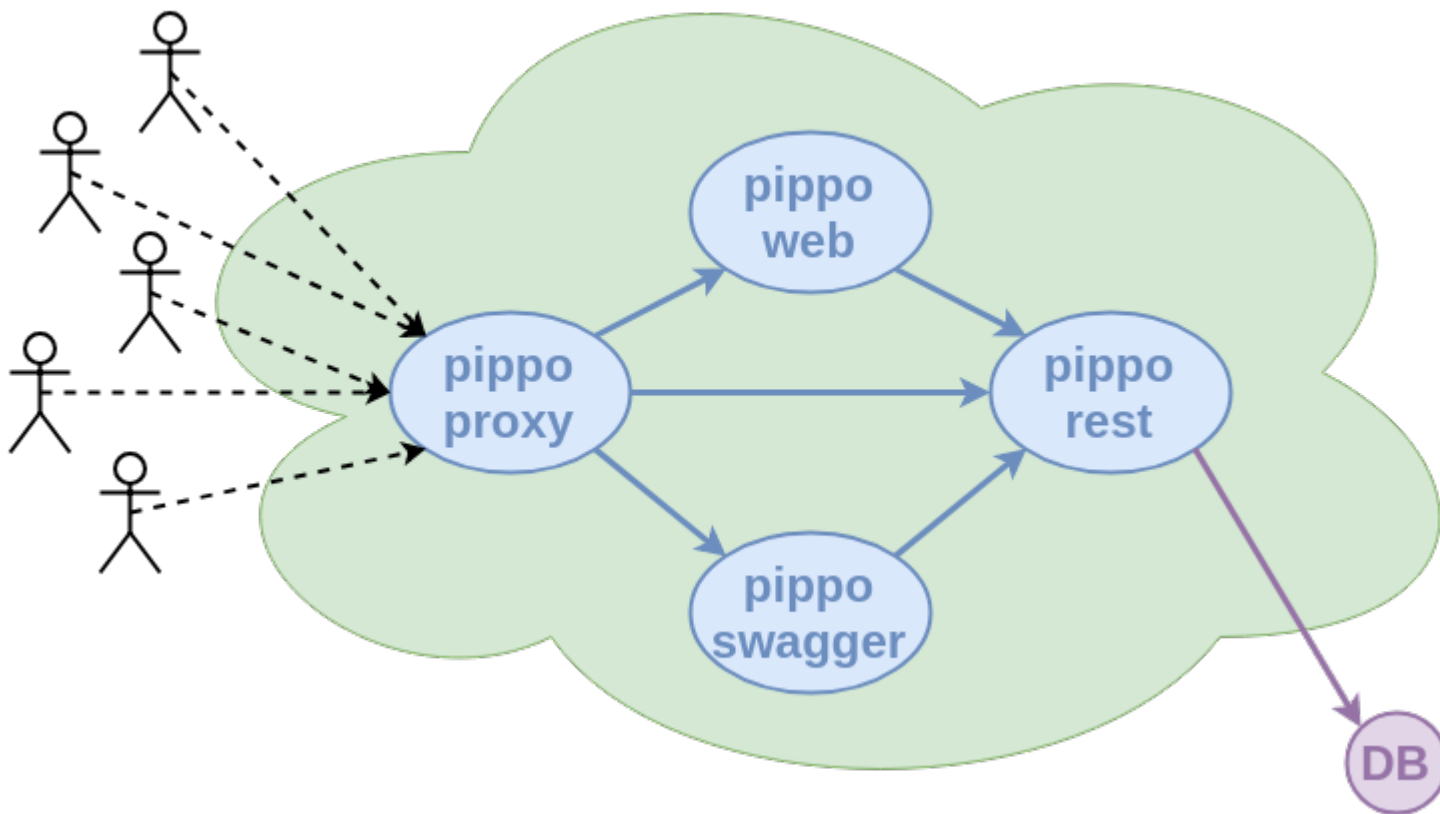


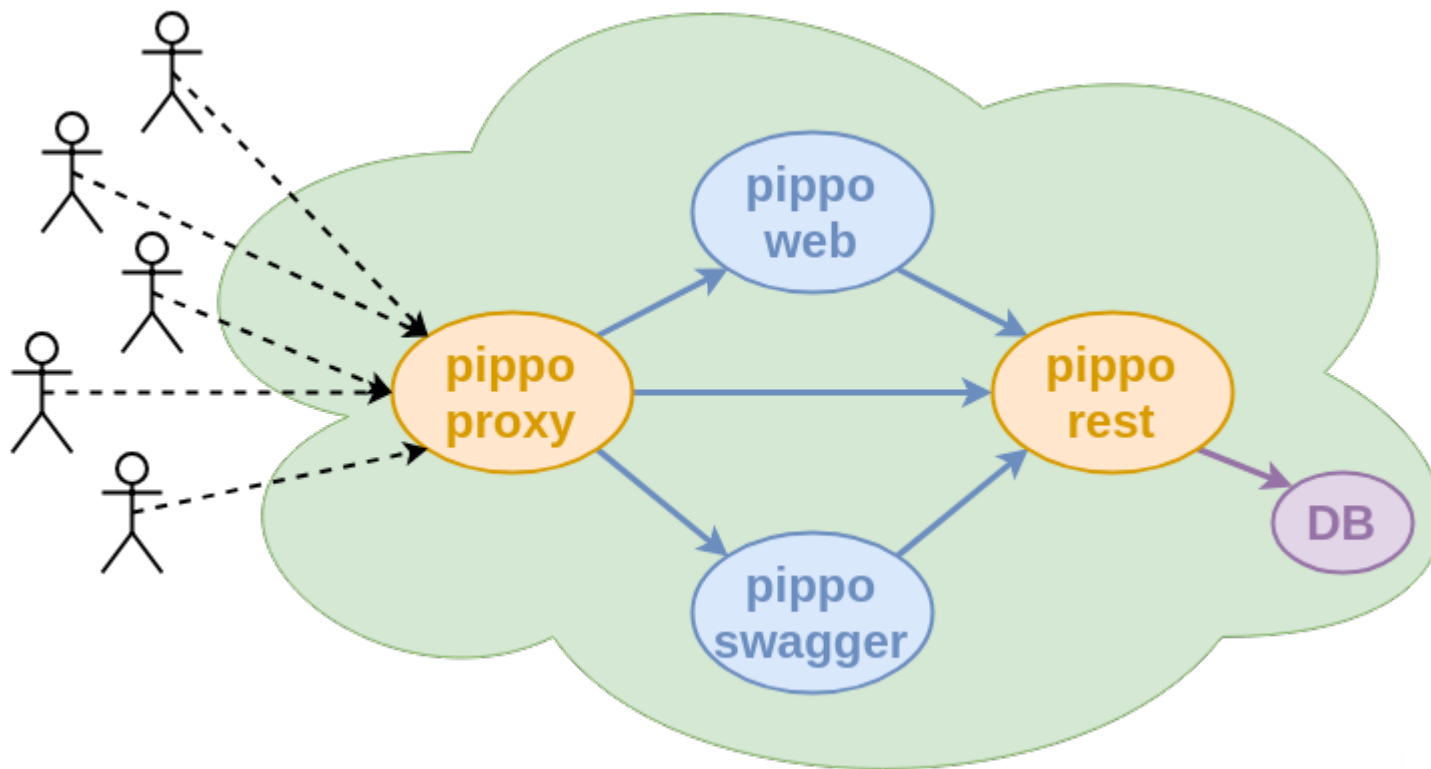
test di integrazione

serve una istanza dedicata dell'applicazione

i dati nel DB devono essere noti

reset dei dati nel DB a ogni esecuzione












Sviluppiamo nuove
funzionalità per l'app.
“pippo”, ma non sono pronte
per i test d'integrazione

Sviluppiamo nuove
funzionalità per l'app.
“pippo”, ma non sono pronte
per i test d'integrazione

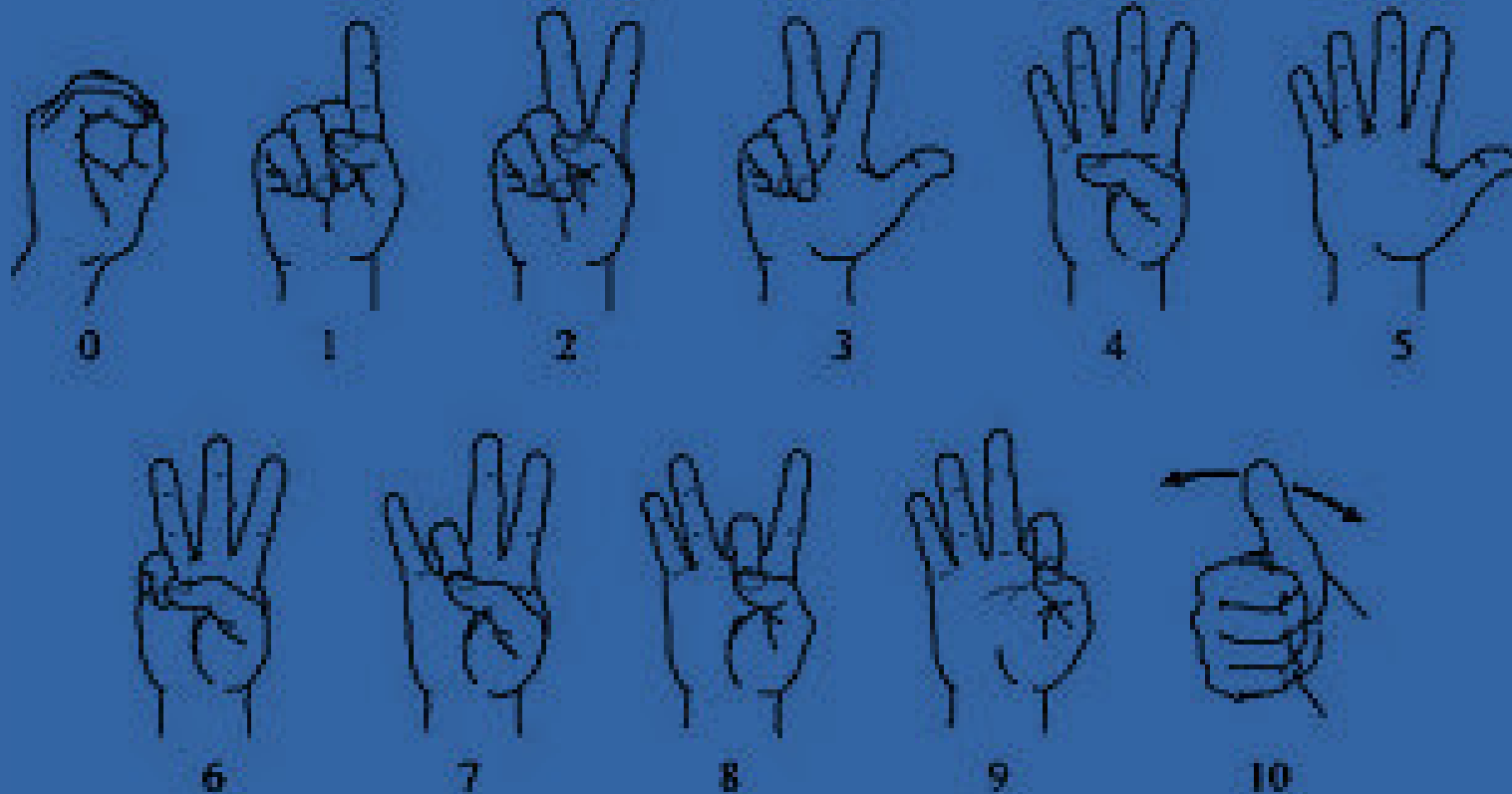
Gitflow

Jenkins multi-branch pipeline



Branches (2)		Tags (0)
S	W	Name ↓
		<u>master</u>
		<u>qa</u>
		<u>develop</u>

counting



-192



0



3



10



11



24



31



112



Andiamo in produzione con docker...
Evviva (forse)!!!



Andiamo in produzione con docker...

Evviva (forse)!!!

- Build fatta dal Jenkins “Corporate”
- Rilasciata sull’artifactory “Corporate”
- Deployed automatico con il sistema di configurazione e gestione “Corporate”
- Unit test, end-to-end test e UAT in ambiente “Corporate”



Due ambienti di build “gemelli”:
2 Jenkins, 2 artifactory (docker-registry)
2 Jenkinsfile e 2 Dockerfile



Due ambienti di build “gemelli”:
2 Jenkins, 2 artifactory (docker-registry)
2 Jenkinsfile e 2 Dockerfile

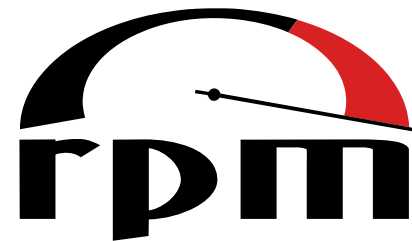
nuova sfida vinta :-)

2 ambienti => 2 variabili d'ambiente, 1 Jenkinsfile, 1 Dockerfile
semplice, no!?

Le nuove installazioni useranno
immagini docker

Quelle vecchie continuano a
utilizzare pacchetti RPM

Il periodo di transizione sarà
lungo... qualche anno





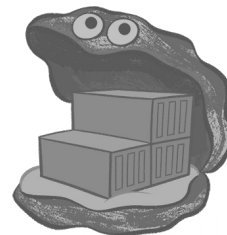
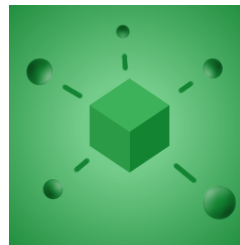
Le nuove installazioni useranno
immagini docker

Quelle vecchie continuano a
utilizzare pacchetti RPM

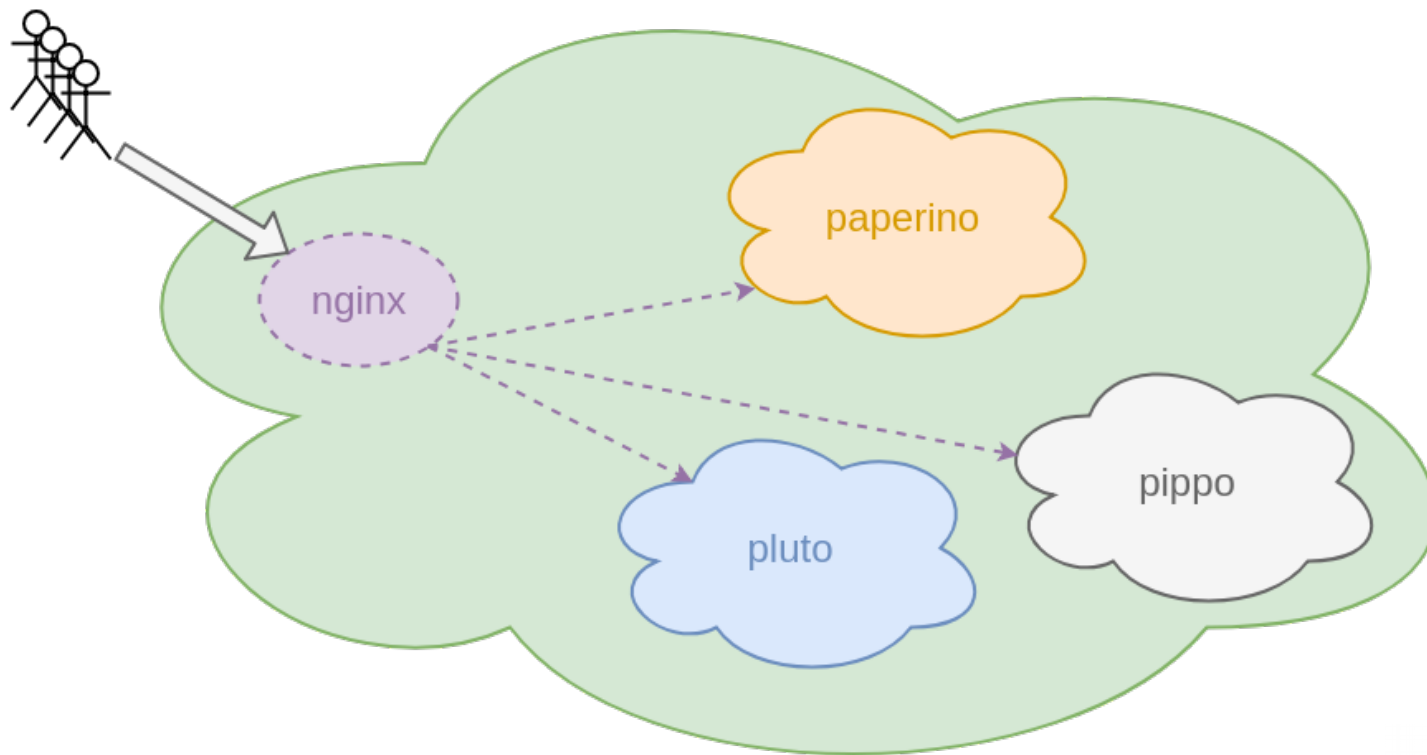
Il periodo di transizione sarà
lungo... qualche anno

Aggiunta di uno step alla
pipeline per creare RPM

Sonatype Nexus OSS 3 al posto
di docker registry



C'è ancora strada da percorrere...



Ci sono altre sfide da affrontare

Creazione automatica certificati SSL

Monitoring dei container docker

Migliorare il processo di configurazione delle applicazioni a livello world-wide per renderlo efficiente e cost-effective

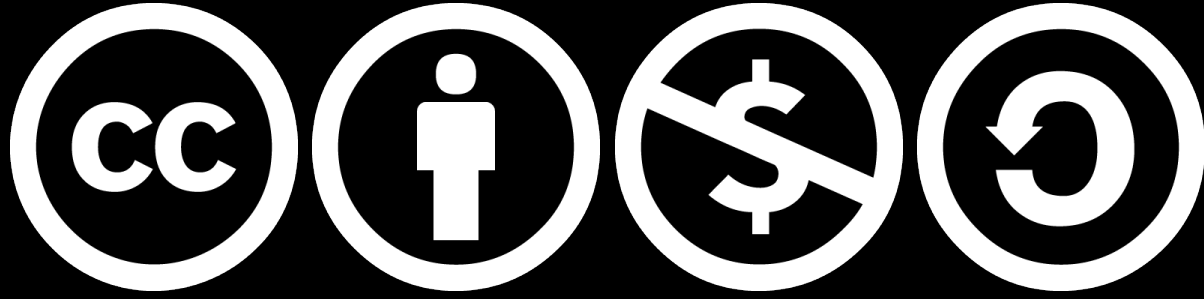
Grazie!

Quando inizi a lavorare con una squadra devi lasciare che il team vada avanti per conto suo. E alla fine devi tutto a loro.

(Michael Schumacher)

Gianni Bombelli
Mini IAD Vimercate
19 maggio 2018

INTR3



Except where otherwise noted, these slides by **Gianni Bombelli** are licensed under a **Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License**:

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Exceptions:

Creative Commons and the double C in a circle are registered trademarks of Creative Commons in the United States and other countries. Third party marks and brands are the property of their respective holders.