L'impératif DevOps
dans un contexte évolutif

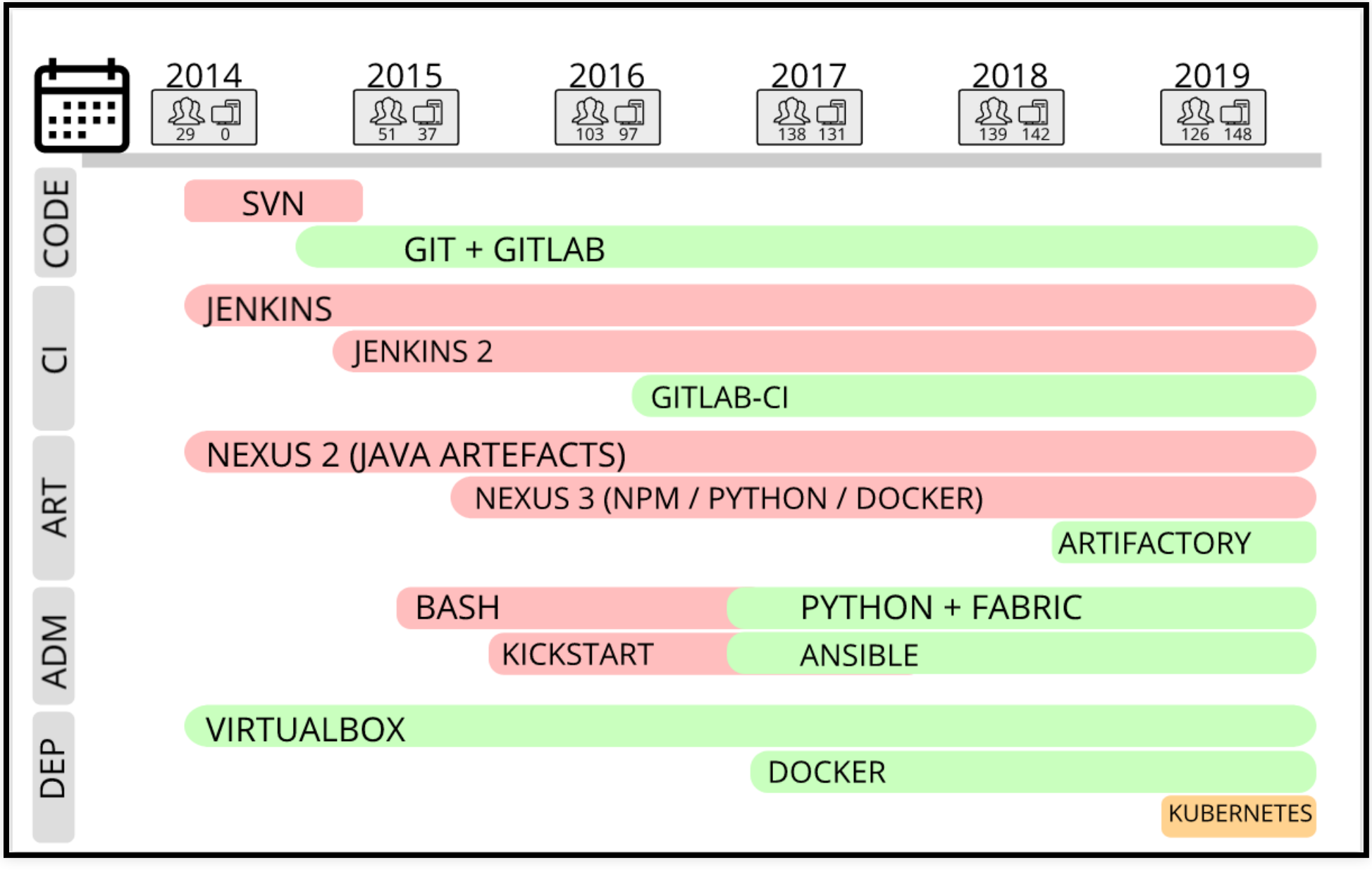
DEFENCE AND SPACE

AIRBUS

Qui suis-je?

- Développement depuis 2003
- Java, python
- Animateur Git/Docker/Python
- DevOps équipe 100+
- Airbus Defence and Space

nicolas.rouviere@airbus.com


2014
29 0
2015
51 37
2016
103 97
2017
138 131
2018
139 142
2019
126 148
CODE
CI
ART
ADM
DEP
SVN
GIT + GITLAB
JENKINS
JENKINS 2
GITLAB-CI
NEXUS 2 (JAVA ARTEFACTS)
NEXUS 3 (NPM / PYTHON / DOCKER)
ARTIFACTORY
BASH
PYTHON + FABRIC
KICKSTART
ANSIBLE
VIRTUALBOX
DOCKER
KUBERNETES

Une transformation réussie

Convaincre Préparer Réaliser
Supporter

CODE

SVN/CLEARCASE to Git

Convaincre

"svn me suffit" / "git c'est compliqué"
Les gens heureux / peur du changement
"c'est pas dans le process" / "combien ça va me coûter"
La peur du coût
"Et mon historique?" / "je vais tout perdre"
La peur d'essuyer les plâtres

git

AIRBUS

Préparer

- Préparer les projets
- Préparer les developpeurs
- Faire une mise en place progressive
- Scripter la migration
- Préparer les workflow (NVIE)

git

AIRBUS

Réaliser

- Golden Day: le jour du ménage
- lancer le script
- Passer SVN en lecture seule

git

AIRBUS

Supporter

- Formation 8h: 400 personnes formées
- Piqûres de rappel sur les fonctions avancées
- Supports ponctuels sur situations délicates
- Support aux autres équipes en cours de migration
- Une mise à jour régulière du workflow

git

AIRBUS

🎓 Git alors?

- +900 projets créés
- Majorité de Gitlab
- Travail multi-sites
- Plus jamais SVN

git

AIRBUS

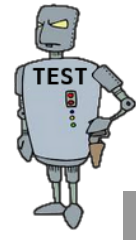
intégration continue

rien => jenkins => jenkins 2 => gitlab-ci

Convaincre

"C'est mieux testé quand c'est moi qui clique"
"Ce que je fais n'est pas testable"
Héritage / peur du changement
"automatiser les tests c'est trop long"
La peur du coût
"mais pourquoi changer?"

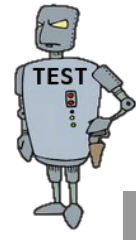
CI



ReConvaincre

CI

Jenkins > *Jenkins 2* > *gitlab-ci*

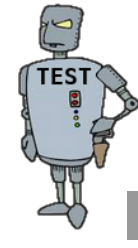


AIRBUS

Préparer

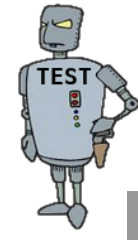
- Préparer l'infra - master/slave
- Préparer les projets
- Préparer les developpeurs à l'ingénierie des tests
- Factoriser les JenkinsFile / gitlab-ci.yml
- Mettre en place des projets "template"

CI



Réaliser

- Mise en place progressive
- Maintient de jenkins2 au début de gitlab-ci

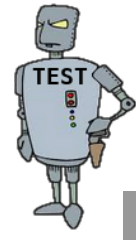


Pipelines

build › *tests U* › *tests I* › *package* › *déploiement*

mvn › *JUnit* › *custom* › *RPMs* › *docker*

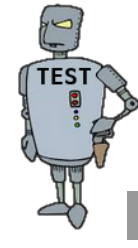
CI



Supporter

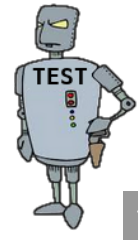
- MAJ gitlab tous les mois
- Upgrade des noeuds CPU/RAM
- Support aux autres équipes en cours de migration

CI



🎓 CI alors?

- ➕ Un investissement permanent mais nécessaire
- ➕ plus de jenkins
- ➖ On a perdu un peu en analyse des failures
- 💡 factoriser grâce à `import`



Gestion des binaires

nexus2 => nexus 3 => artifactory

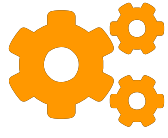
Convaincre

"mais pourquoi changer?"
"combien ça va me coûter"
"Et mon historique?" / "je vais tout perdre"

Préparer

- Penser ses repository par projet
- Sizer la machine correctement
- Faire de la doc sur comment utiliser les repositories
- Mettre en place des règles de nommage fortes!

AIRBUS



Réaliser

- `docker-compose up`

Supporter

- Créer des repositories quand un projet démarre
- Surveiller l'espace disque
- Faire du ménage

AIRBUS

🎓 Artifactory alors?

- ➕ Artifactory c'est bien
- ➖ Difficile: se séparer des anciennes zones
- ➖ Difficile: l'authentification à posteriori
- ➖ Difficile: Faire respecter les règles de nommage
- 💡 Penser loin en découpage de repo

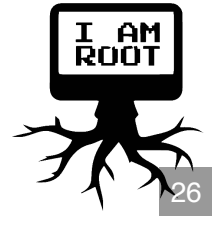
AIRBUS

SYSOP

Kickstart + bash => python + ansible

Convaincre

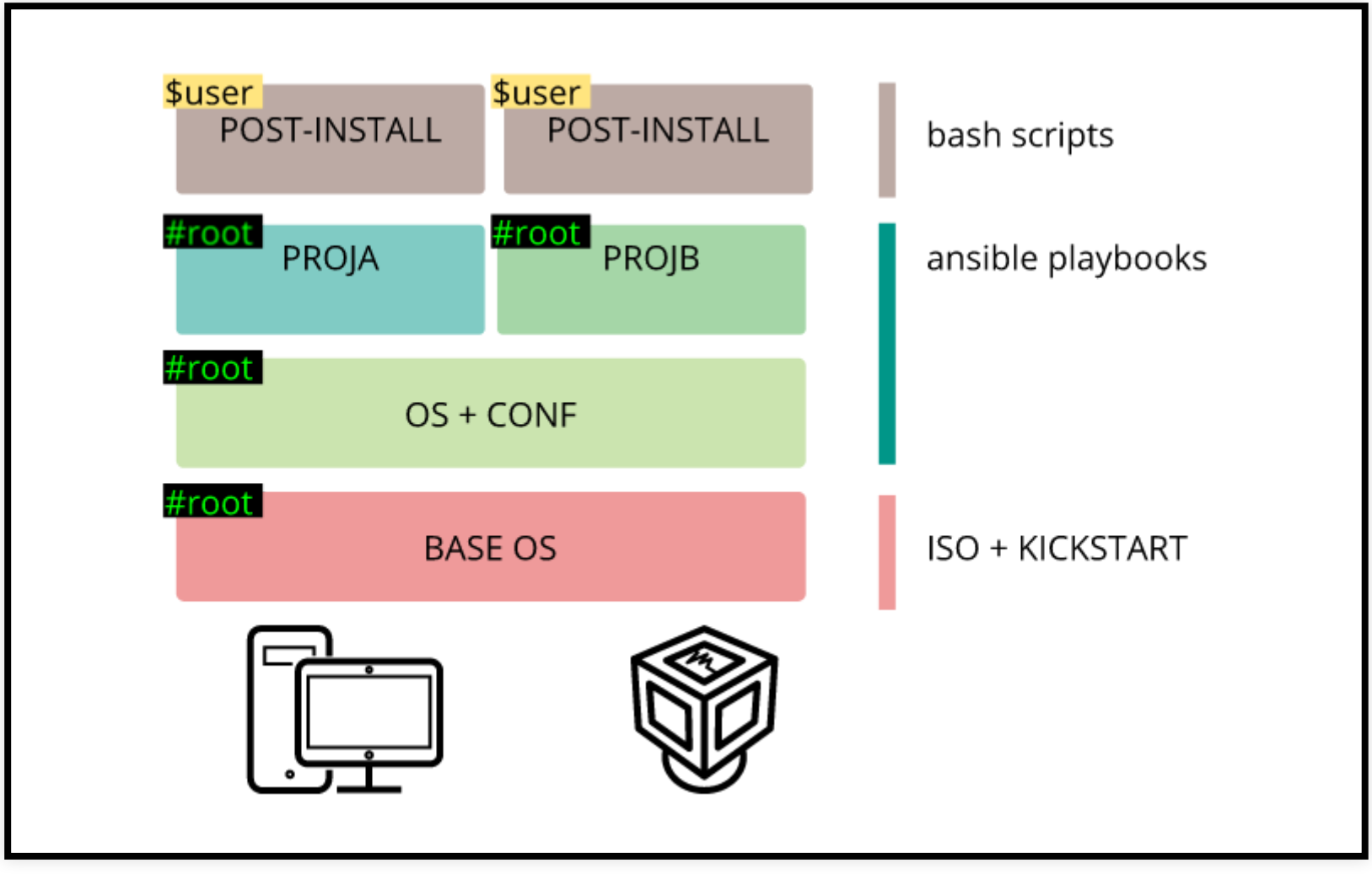
"Personne ne peut reprendre ton ansible!"
"Python, c'est pas serieux, BASH oui"

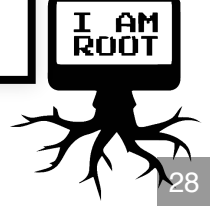


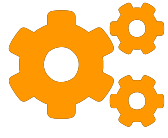
Préparer

- Penser playbooks de manière evolutive
- Limiter l'hétérogénéité du parc
- Faire de la doc sur comment utiliser les playbooks
- Faire de la doc sur comment opérer le système
- Penser au "besoin d'en connaître"

Préparer

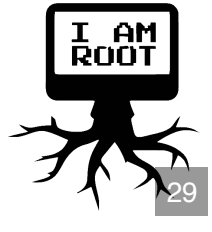

$user
POST-INSTALL
$user
POST-INSTALL
bash scripts
#root
PROJA
#root
PROJB
ansible playbooks
#root
OS + CONF
#root
BASE OS
ISO + KICKSTART


I AM
ROOT



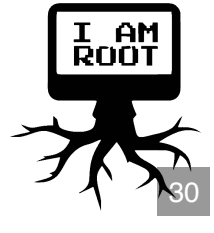
Réaliser

```
ansible-playbook doit.yml
```



Supporter

- Suivre les changements d'OS
- Mettre à jour ses playbooks au lieu de faire des fixs crado
- Rester solide face aux demandes non conformes



Ansible alors?

- ➕ facile à prendre en main
- ➕ investissement largement rentabilisé
- ➕ 2 OS - 3 Hardware
- ➖ les dev ont peu de connaissances système
- ➖ je n'aurais pas du donner le sudo
- ➖ dur de bosser en offline
- 💡 AWX/Tower!
- 💡 authentification centralisée!

DEPLOY

VBOX + DOCKER + KUBERNETES

Convaincre

"machine virtuelle? je préfère un vrai système"
"docker on a pas de recul en operationnel"
"docker c'est une faille de sécurité"
"kuber-quoi ?? Le cloude?"

AIRBUS

Préparer

- Bien choisir ses images de base
- Contruire des images de base métier
- Versionner / publier par intégration continue
- Bien sizer pour éviter l'overdose

AIRBUS

Supporter

- Formation / introduction à docker
- Support à la réalisation des charts helm
- Formation avec des entreprises spécialisées
- Veille techno

🎓 la virtualisation alors?

- ➕ virtualbox + vagrant
- ➕ docker + docker-compose
- ➖ confusion VM/Container
- ➖ TRES dur de bosser en offline
- ➖ Difficile: Le virage CLOUD
- 💡 Faire des templates (cookiecutter)
- 💡 Prévoir une infra LARGE
- 💡 Se faire former pour éviter les erreurs!

AIRBUS

6 ans!

🎓 le DevOps alors?

- ➕ C'est passionnant
- ➕ Ca bouge tout le temps
- ➕ Ca permet de changer d'echelle
- ➖ Difficile: la peur du changement
- ➖ Difficile: apprendre à se tromper
- ➖ Difficile: sans le net
- 💡 S'adapter au contexte/criticité de l'application
- 💡 Il faut des gens dédiés aux compétences multiples
- 🎓 💡 Il faut un support de la hierarchie

The Red Queen Effect

"Now, here, you see, it takes all the running you can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!"

Alice's Adventures in Wonderland -Lewis Carroll

« Ici, vois-tu, on est obligé de courir tant qu'on peut pour rester au même endroit. Si on veut aller ailleurs, il faut courir au moins deux fois plus vite que ça ! »

DEVOPS TOOLS