Quel modèle de plateforme calcul pour la R&D ?

DecaLog { ComputeOps, Reprises }

30 mars 2026 @CC-IN2P3

Pierre Aubert (LAPP),

Fabrice Jammes (LPCA) et

Richard Randriatoamanana (Subatech)

Wishlist

*Développer à l'IN2P3 une plateforme de calcul à la fois **agile, frugale et sécurisée** dédiée exclusivement aux développements de codes scientifiques dans une infra cloud performante, convergée et managée par un ecosystème multi-ordonnanceurs Kubernetes/Slurm avec une gestion efficace et supervisée des ressources HPC/HTC, tout en offrant aux développeurs un environnement de simulation et de pré-production normalisé pour faciliter la livraison logicielle par workflow (CI/CD) et le portage et l'optimisation de code en vue d'un déploiement en production contrôlé et d'une montée en scalabilité garantie.*

Notre point de vue (pour la R&D)

Le constat (aujourd'hui)	La vision (demain)
Accès complexe : ressources méconnues et parfois indisponibles et inadaptées	**Standardisation :** un accès uniforme à tous les étages (workflows, outils, modèles, etc.)
Accompagnement fragile : expertise de proximité partielle voire parfois absente.	**Coopération efficace :** des outils partagés et des REX pour aligner les équipes (expertise et soutien)
Manque d'agilité : pas d'environnement (ou *custom-made*) *userland* de tests et de simulation (sandbox / twin / mock).	**UserLand R&D :** un espace configurable dédié au design et aux POC/tests (eg. *google codelabs*) pour une continuité dans la livraison logicielle (CI/CD)
Usage rigide : l'essentiel des plateformes (T1/T2) de l'ESR est sur le modèle full-production	**Flexibilité :** Multi-ordonnanceurs pour plus de fluidité.

Backup


On-site
IaaS
CaaS
PaaS
FaaS
SaaS
Data
Apps
Runtime
Container
Middleware
OS
Hypervisor
Servers
Storage
Networking
Functions
Block Logic
Scalability
Parallelization
Provisioning
Err. Handling
Managed by the user
Managed by the provider
Managed by the user and the provider
Configurability
Agility

Cloud native tools and apps...

- Automation & Configuration
  - Automating software delivery : dagger.io
  - OpenTofu (Terraform-forked) for an on-cloud deployment (Infra-as-Code)
- Kubernetes-as-an-OS : *Talos* / *FlatCar* / *Rancher* / *RedHat-OpenShift*
- Manage bare-metal by Kubernetes with Metal3
- **Observability and Analysis** (*opentelemetry*, *fluentd*)
- Manage VMs and Containers with KubeVirt.io
- Manage native kube apps with Operator Framework
- Build your cloud native registry (*harbor* / *RedHat-Quay.io*)

Notre constat / les faits

- encore des ressources méconnues dans notre paysage ESR (FR+EU)
- c'est jamais le bon moment (panne, lenteur, pas de ressources, en attente, etc.)
- accompagnement d'expertise tech de proximité timide (non identifié) voire parfois absent
- environnement de bench pour faire de la simulation et des tests inexistant ou *custom-made* (testbed / digital twin / mocking)
- cc|htc-hpda + idris|hpc-ai > prod/run usage
  - userenv for R&D (design/poc/proto/test) ?

Standardisation à tous les étages

- L’intégration d’un multi-ordonnanceurs de workload
- Simplification des mises à jour et déploiements continus et non disruptifs, préservant ainsi la stabilité de l’infrastructure hébergeuse ;
- Garantie des environnements uniformisés;
- Favorisation pour une meilleure coordination entre les équipes techniques des différents sites dans l’adoption de pratiques et d’outils standardisés et partagés.

Recherche d’un équilibre entre mutualisation et agilité

- un modèle de plateforme (T1/T2) dédié à la R&D ? (*optimized userland optimized, secured, powerful and agile*)
- un modèle pyramidal pour une montée en scalabilité progressive et validée (portabilité et sécurité);
- userdev → infra-dev → infra-preprod → infra-prod
- vers une proximité géographique d’accompagnement avec les experts (*task-force for an intensive and focused work/mission*) in-situ et de visu (F2F)

Sécurité et gouvernance

- Approche adaptée par contexte.
- R&D : *security by proactive monitoring*

  vers de l'EDR AI-driven et quarantaine/isolation dans une DMZ
- Apps (CSAN) : *security on workflow*

  avec des scans image (sbom-based) en amont pour détecter des vulnerabilités type Zero-Day
- Production: *security full stack and compliance certified*

  avec des scans et des audits réguliers obligatoires en amont

Flexibilité technologique

Grâce aux standards bien établis (*Kubernetes, Slurm, OCI, SBOM, Registry, Runners for CI/CD*), on pourrait penser:

- une rotation facilitée des architectures matérielles (CPU/ARM, GPU/TPU)
  loop { switch + test/mock + debug } → release
- une interopérabilité des systèmes entre plateformes
- une utilisation optimisée des ressources et des environnements “*userland*” par les développeurs R&D pour leurs POCs/TestBeds.

Vers quel modèle de plateforme R&D ?

- L’approche par montée en charge progressive…
- Un idéal: vers des vers plateformes hybrides
  Dev/R&D → intégration → production
- Diversification vs. concentration des ressources
- Les économies d’échelle justifient aujourd’hui le maintien des plateformes existantes (T1/T2) avec la labellisation et l’homologation de sécurité
  - climatisation, PUE, redondance électrique, sécurité des sites, etc.
- Un arbitrage nécessaire ? efficacité énergétique/économique et résilience
  - vers l’hybridation / la convergence

#GCPSketchnote

@PVERGADIA

THECLOUDGIRL.DEV

08.11.2021

Wait... what is Cloud again?

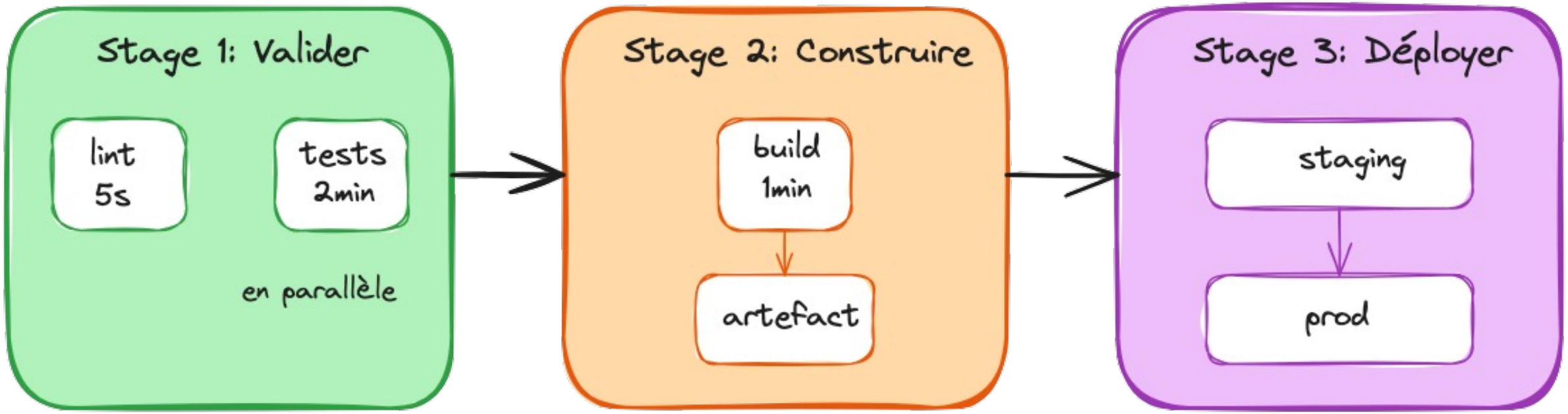
TRADITIONAL ON-PREMISES	INFRASTRUCTURE AS A SERVICE (IaaS)	CONTAINERS AS A SERVICE (CaaS)	PLATFORM AS A SERVICE (PaaS)	FUNCTION AS A SERVICE (FaaS)	SOFTWARE AS A SERVICE (SaaS)
Data & Configurations	Data & Configurations	Data & Configurations	Data & Configurations	Data & Configurations	Data & Configurations
Application Code	Application Code	Application Code	Application Code	Application Code	Application Code
Scaling...	Scaling...	Scaling...	Scaling...	Scaling...	Scaling...
Runtime	Runtime	Runtime	Runtime	Runtime	Runtime
OS	OS	OS	OS	OS	OS
Virtualization	Virtualization	Virtualization	Virtualization	Virtualization	Virtualization
Hardware	Hardware	Hardware	Hardware	Hardware	Hardware

You Manage

Cloud Provider Manages

https://cutt.ly/5tPZi6Me

Autonomie d’une pipeline

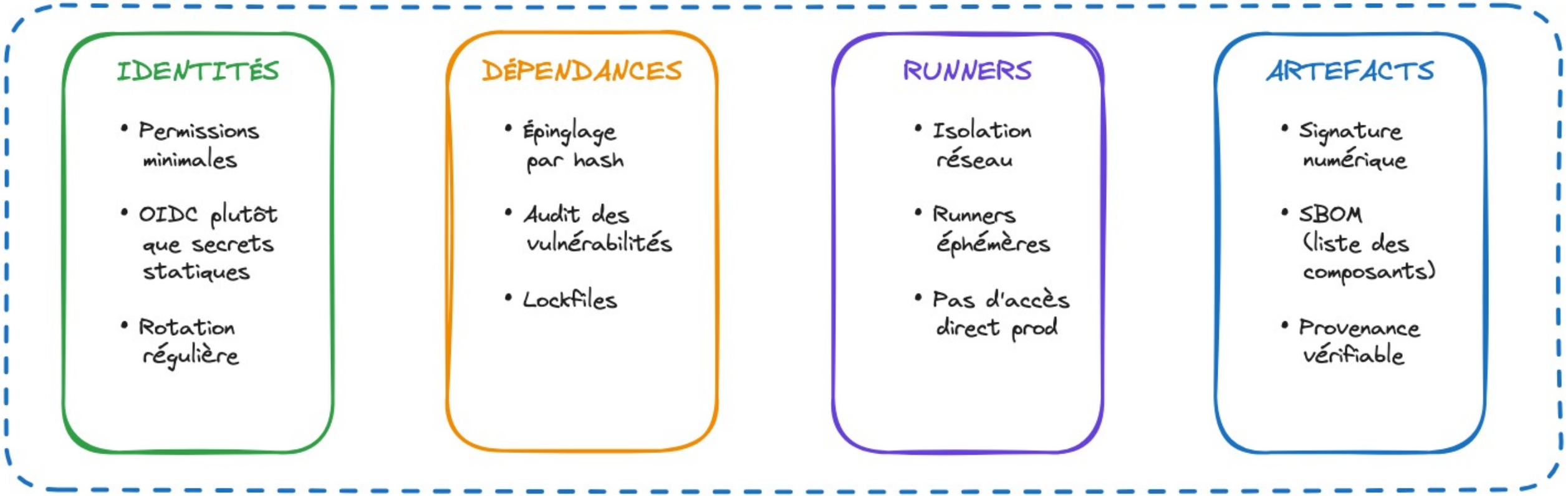


source: https://cutt.ly/RtPPACGU

Vocabulaire :

- Job = une tâche (lint, test, build, deploy)
- Stage = un groupe de jobs (validation, construction, déploiement)
- Runner = la machine qui exécute les jobs
- Artefact = ce qui est produit (image Docker, binaire, package)

Les 4 surfaces de contrôle



source: https://cutt.ly/jtPPIAHS