Dauphine | PSL
UNIVERSITÉ PARIS

DevOps

Introduction

Antoine Balliet
antoine.balliet@dauphine.psl.eu

Avant de commencer

- Matériel requis : un ordinateur par binôme
- Expérience en DevOps parmi les étudiants ?
- Expérience en dev, outils ?

Présentation

Organisation du cours

- 6 séances / 18h de cours au total : 5h de cours / 13h de TP
- 1 TP en autonomie évalué ~2h (Q/A avant)
- Slides de cours disponibles sur GitHub

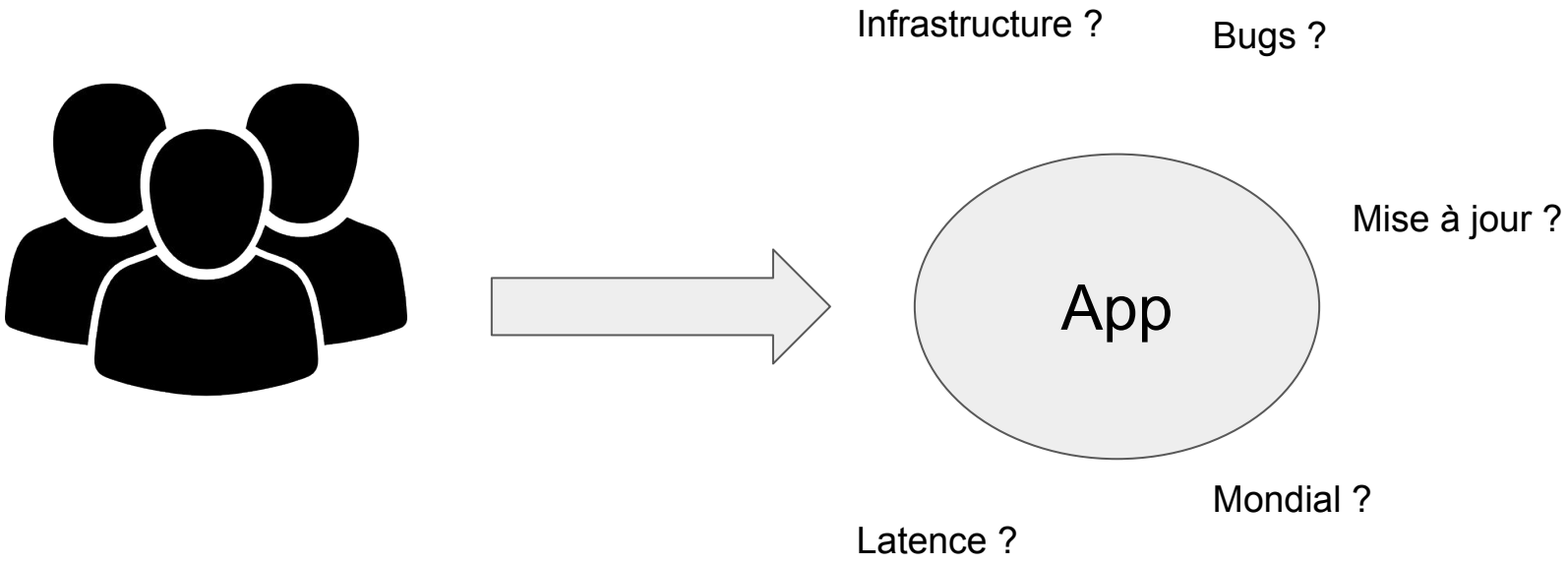
Evaluation

- Note de TP (50%) : 30 Janvier
- Devoir sur table (50%) : 6 Février

Prérequis

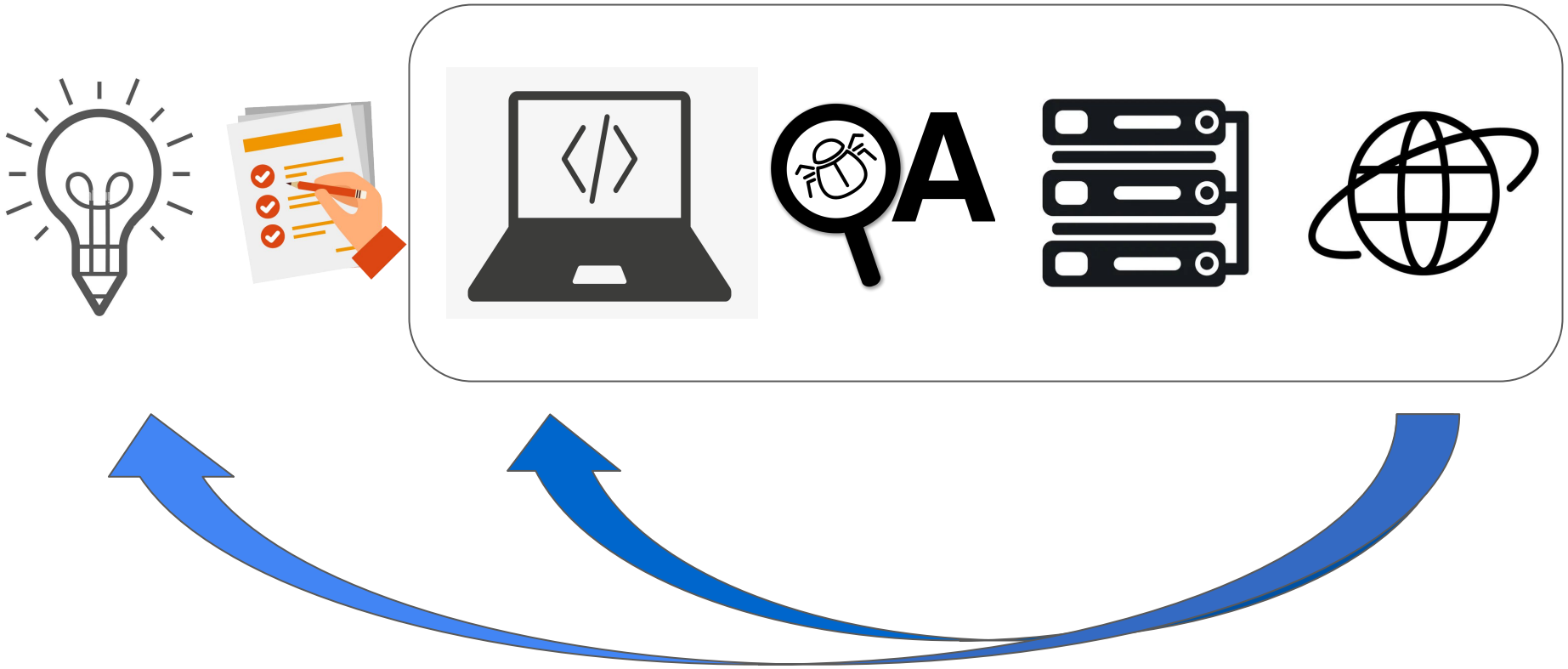
- Pas de développement applicatif en soi mais connaissance shell / bash requis
- 1 ordinateur pour 2
- Curiosité : n’hésitez pas à chercher sur internet :)

Enjeux majeur du secteur : déploiement en production

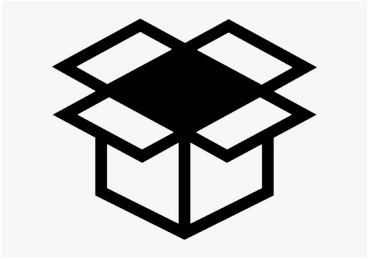
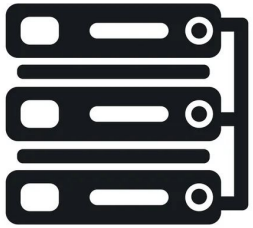



QA

- Mesure et amélioration continue des performances
- Nouvelles fonctionnalités
- Vélocité d’exécution



Processus de livraison d’une application


- Installer les dépendances
- Tests unitaires / fonctionnels
- Déployer l'application
- Vérifier son comportement
- Adapter les ressources à la charge
- …

Pourquoi ce cours ?

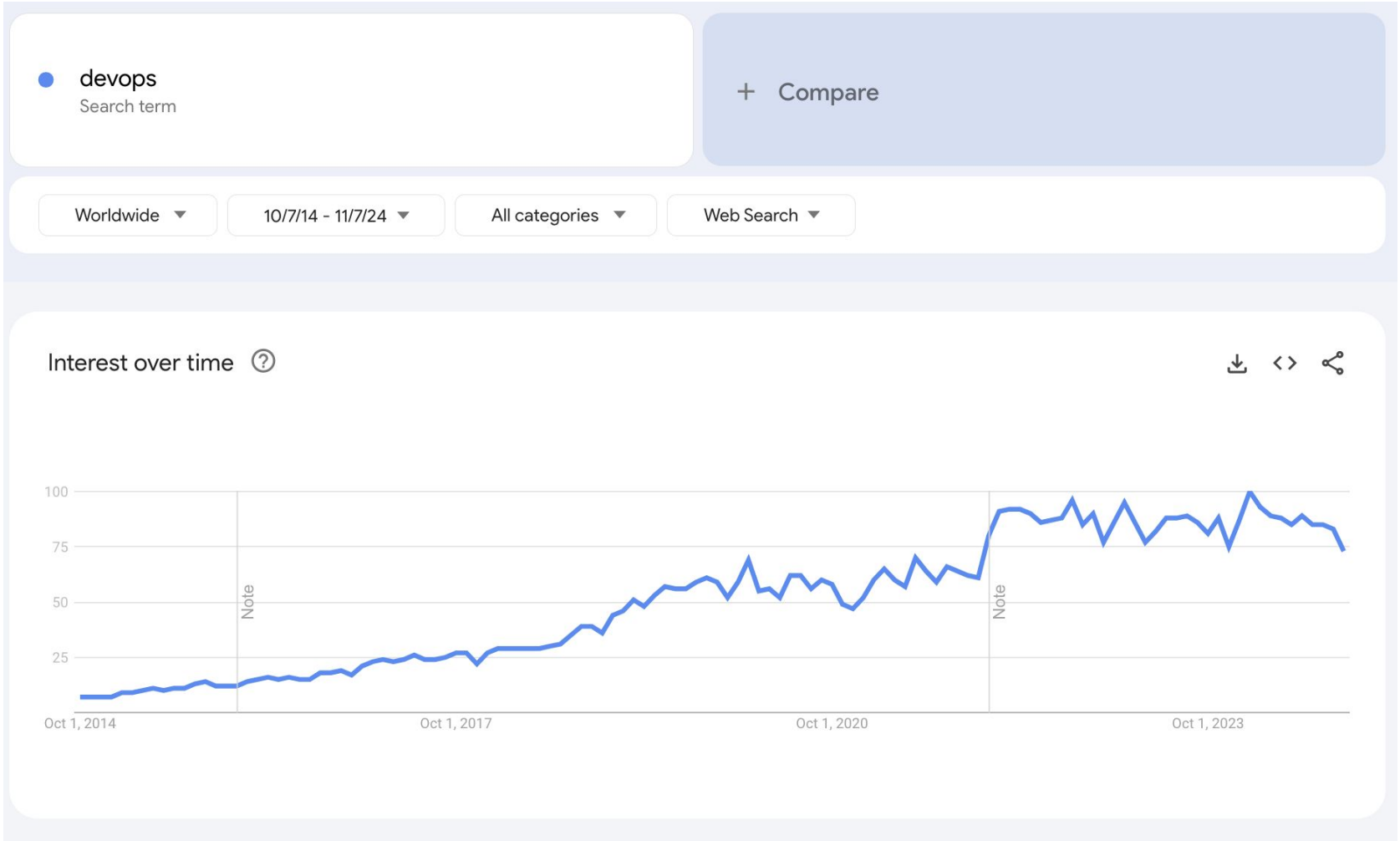
Initier aux outils et pratiques modernes pour le déploiement et la gestion des applications à travers la culture DevOps

Démocratiser notre usage et créer une confiance dans le logiciel

“$5,600 per minute to nearly $9,000 per minute for average”

“For eCommerce giant Amazon, whose entire business model relies on uptime, estimated costs are around $13.22 million per hour.”

https://www.atlassian.com/incident-management/kpis/cost-of-downtime



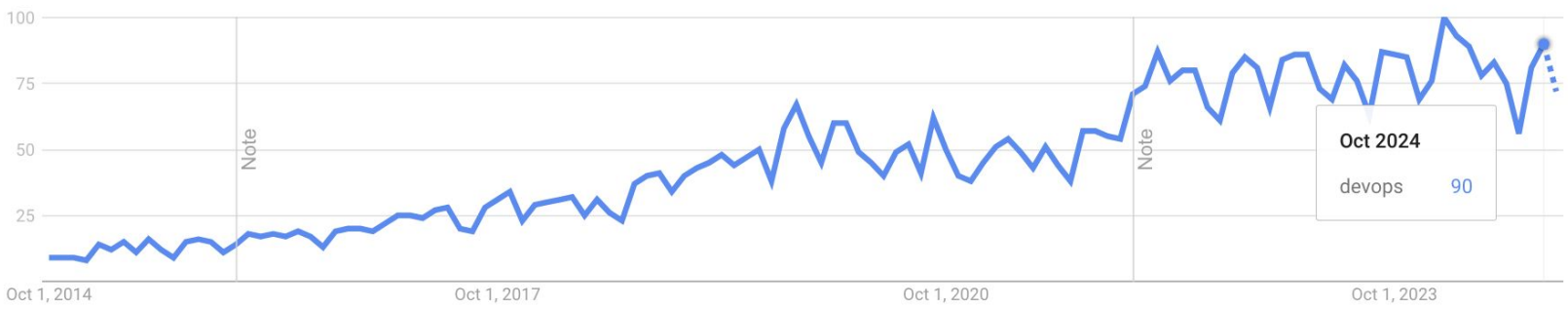
https://trends.google.com/trends/explore?date=all&geo=FR&q=devops&hl=en

devops
Search term

+ Compare

France | 10/7/14 - 11/7/24 | All categories | Web Search

Interest over time

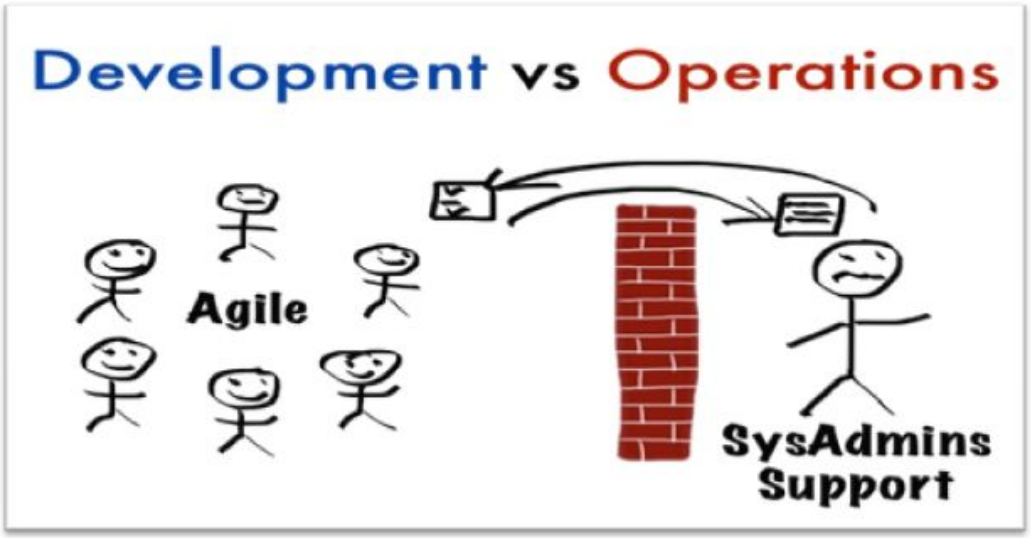


https://trends.google.com/trends/explore?date=all&geo=FR&q=devops&hl=en

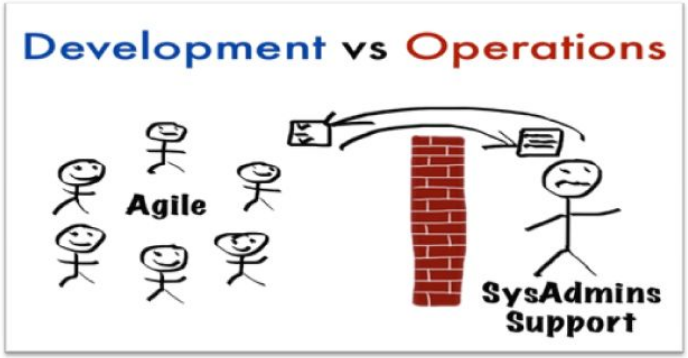
Avant le “DevOps”

2000 - 2007:

- Applications se complexifient (millions de ligne)
- Investissement propre en infrastructure
- Dépendance croissante à du code existant externe



Relation entre Dev et Ops



Dev: équipes de développeurs logiciels
Ops: en charge de la mise en production des produits

Antagonisme fort:

- Dev: Modifications aux moindres coûts, le plus vite possible
- Ops: Stabilité du système, qualité

Automatisation et responsabilisation sont au cœur de l'approche DevOps

Approche DevOps

“You build it, you run it” (2006) - Werner Vogels
(CTO & VP @Amazon)

“The most powerful tool we have as developers is automation”
Scott Hanselman (VP @Microsoft)

“Deployment celebrations should be about the value of the new features, not joyful relief that nothing went horribly wrong”
Rebecca Parsons (CTO @Thoughtworks)

Notions abordées du cours

Développement collaboratif : CI/CD, versioning, git, Cloud Build

Virtualisation, conteneurisation et orchestration : docker, kubernetes

Cloud providers / hyperscalers : concepts du cloud, services GCP

Infrastructure as Code : cas d'usage, terraform

DevOps

Ensemble de techniques et d'outils facilitant le passage du développement à la production.

Bien plus que ça:

- Modèle de fonctionnement de l'entreprise
  - Impliquant tous les maillons de la chaîne (RHs, finances, etc.)

- Modèle d'interactions entre les équipes
  - Intégration du retour sur expérience
  - Une "culture"

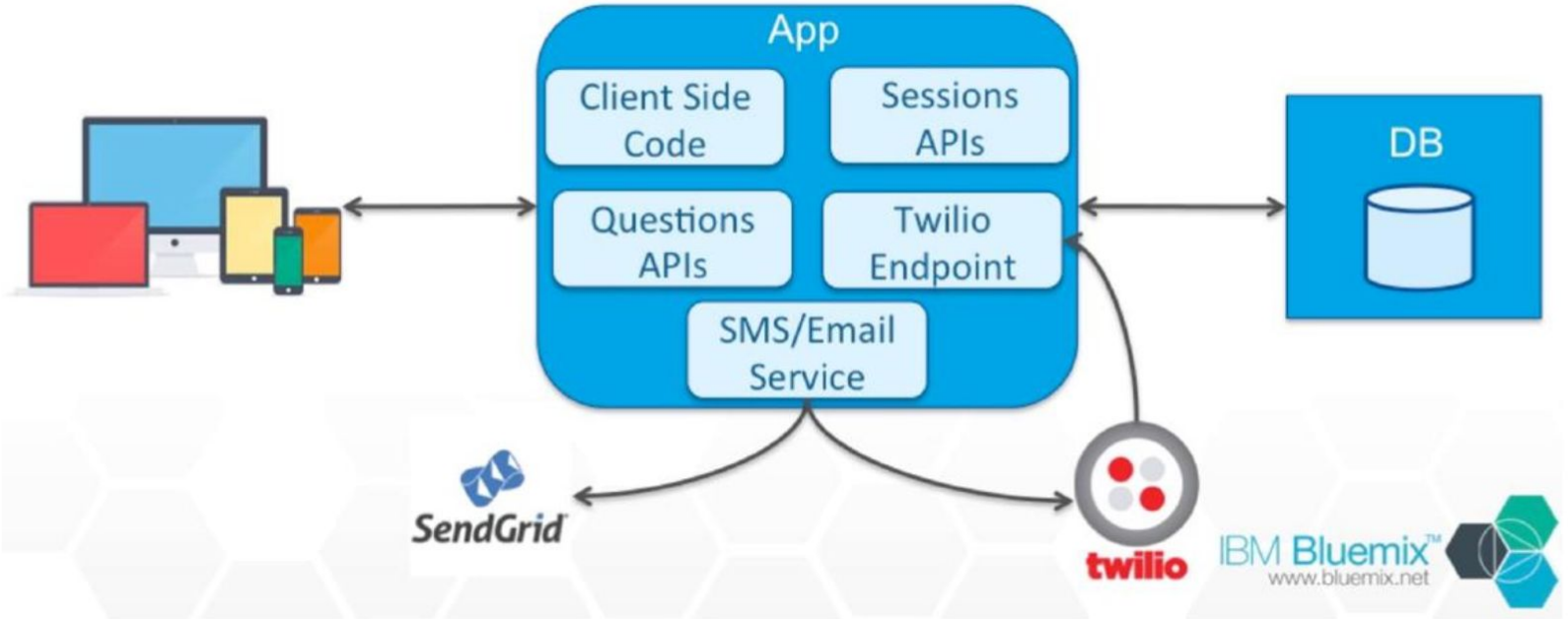
Dauphine | PSL
UNIVERSITÉ PARIS

DevOps

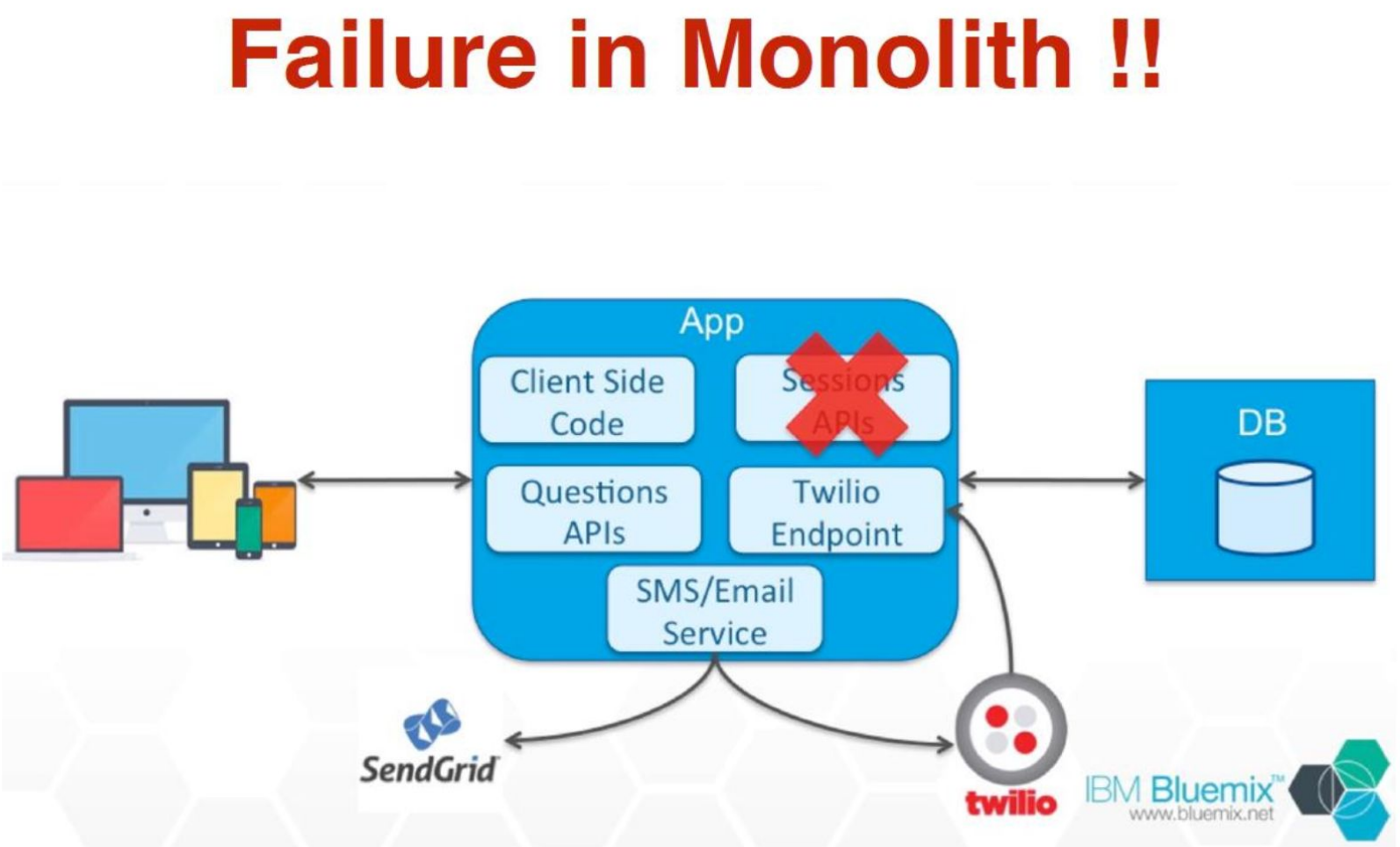
Cours 1 : Docker

Antoine Balliet
antoine.balliet@dauphine.psl.eu

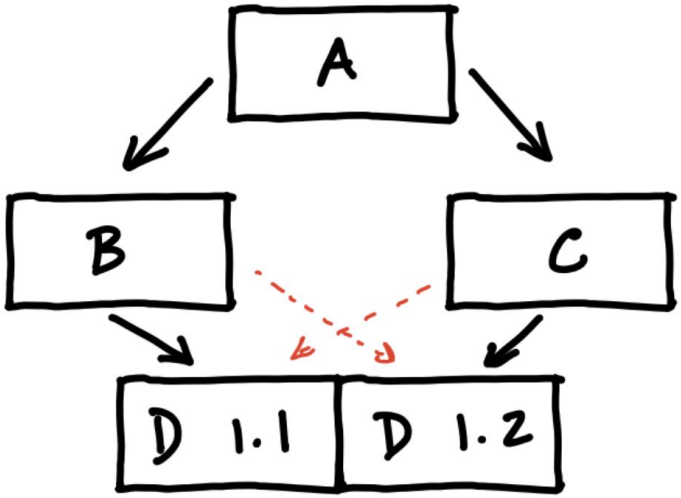
Isolation des applications



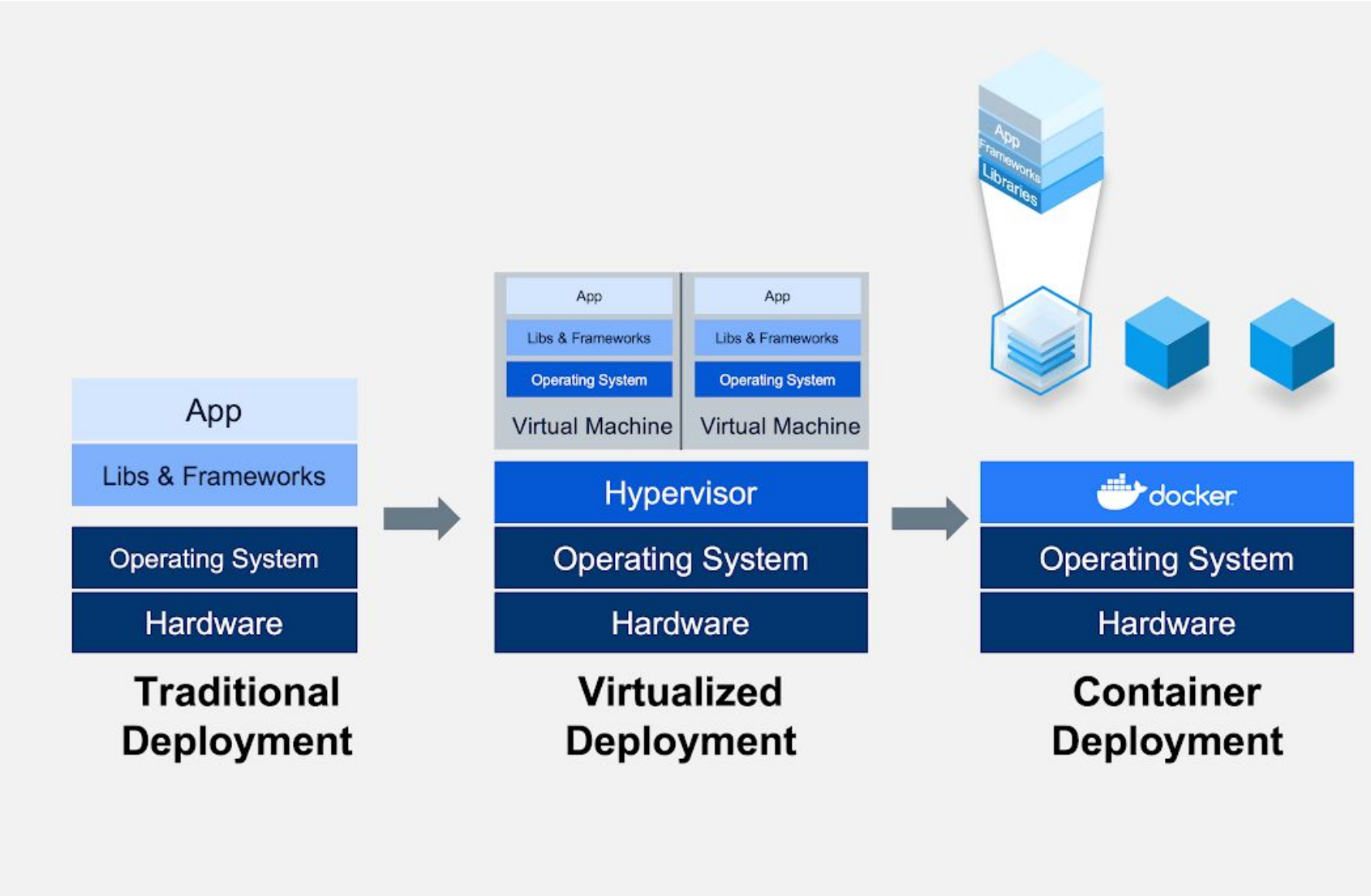
Isolation des applications



Gestion des dépendances

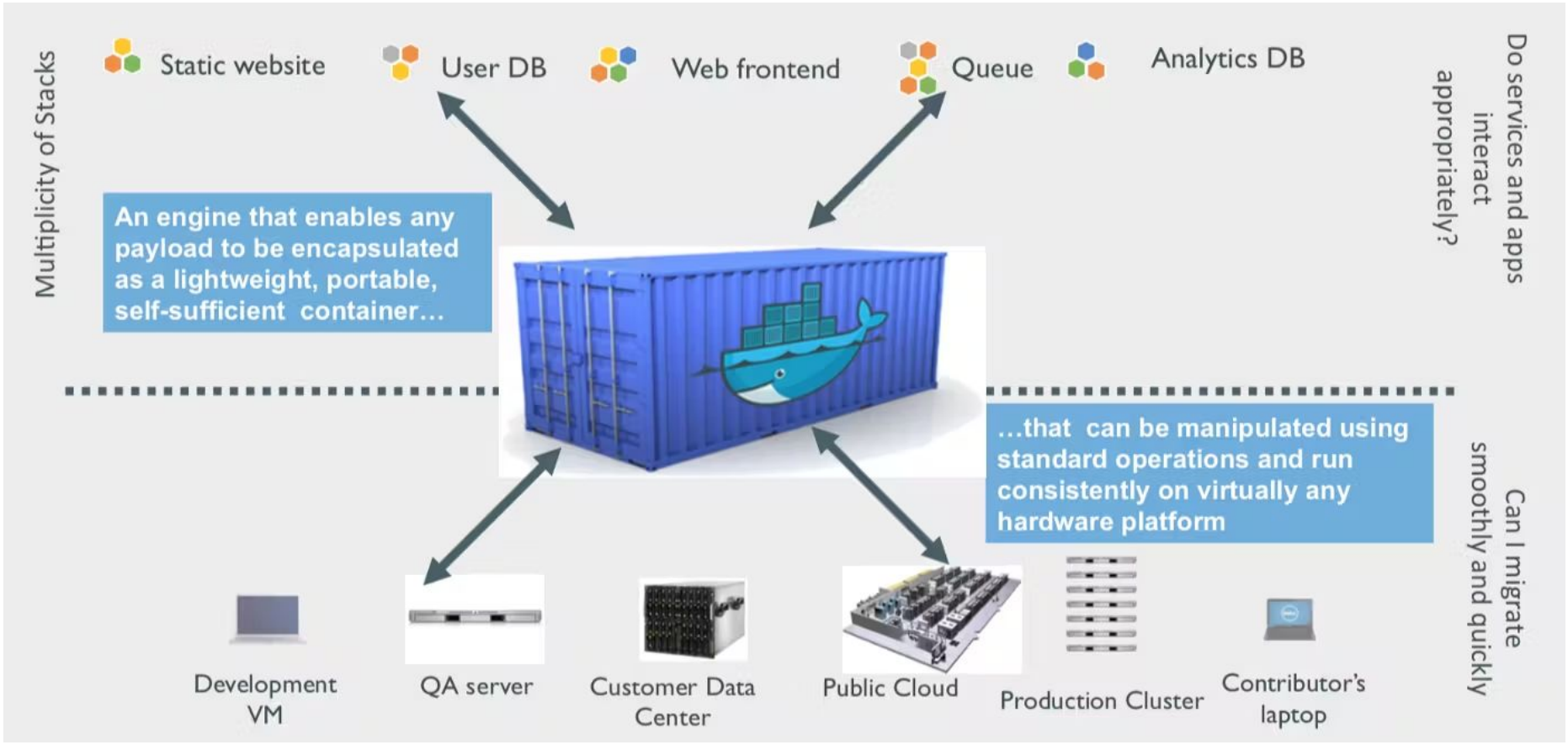


Les déploiements



Conteneurisation

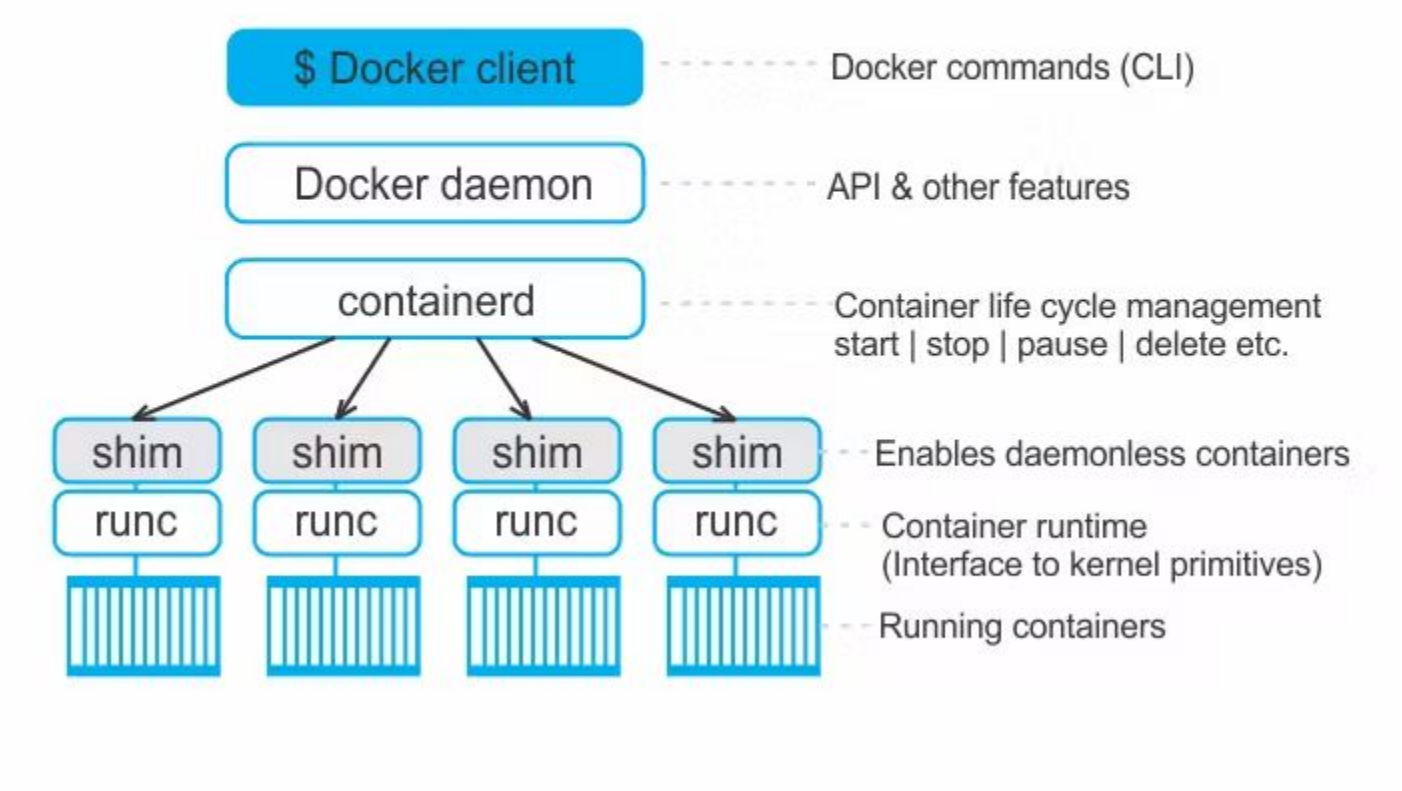
Conteneurisation



Docker

- **open source** (Apache 2.0)
- crée par la startup DotCloud (renommée depuis Docker) fondée par deux français anciens de l'Epitech.

Docker


$ Docker client
Docker commands (CLI)
Docker daemon
API & other features
containerd
Container life cycle management
start | stop | pause | delete etc.
shim
shim
shim
shim
Enables daemonless containers
runc
runc
runc
runc
Container runtime
(Interface to kernel primitives)
Running containers

Docker

- Uniformiser les environnements : 10 devs travaillent sur multiples OS (Ubuntu, Windows, MacOs, ...)

- Création des environnements locaux

- CI/CD: Création des espaces isolés pour valider les chaînes de tests/déploiements

Docker

- Stateful vs Stateless
- Immutabilité: un conteneur n'a pas pour vocation de stocker des données.
- Définition des ressources

“Application instances should be treated more as cattle”.
Twelve factor apps.

https://developer.ibm.com/articles/creating-a-12-factor-application-with-open-liberty/

Pour aller plus loin

- https://docs.docker.com/get-started/
- https://12factor.net/ : twelve factor apps. Principes généraux d'architecture pour les applications distribuées

Plateformes utilisées

- Pour les TPs nous utiliserons Google Cloud Platform & GitHub

  - Cloud Shell : Interface simple pour développer sur le cloud
  - Services managés :
    - Artifact Registry
    - Cloud Run
    - Cloud Build
    - Kubernetes Engine
    - …

=> 50$ de crédit par étudiant

Adresse de la promo: ***A5ASIT-100.2024@groupes.dauphine.eu*** ?