*Ops, c'est quoi le rapport avec R ?

> DevOps, DataOps, MLOps, StatOps, AIOps et consorts: quelques tentatives de définitions et quel rapport avec R ?

#person("Diane", "Beldame", email = "diane@thinkr.fr", role = c("aut", "cre"))

```
> quote(”In previous economic eras businesses created values by moving atoms. Now they create value by moving bits”

- Jeffrey Snover, chief architect Azure @Microsoft)
```

	1970s–1980s	1990s	2000s–Present
Era	Mainframes	Client/Server	Commoditization and Cloud
Representative technology of era	COBOL, DB2 on MVS, etc.	C++, Oracle, Solaris, etc.	Java, MySQL, Red Hat, Ruby on Rails, PHP, etc.
Cycle time	1–5 years	3–12 months	2–12 weeks
Cost	$1M–$100M	$100k–$10M	$10k–$1M
At risk	The whole company	A product line or division	A product feature
Cost of failure	Bankruptcy, sell the company, massive layoffs	Revenue miss, CIO's job	Negligible

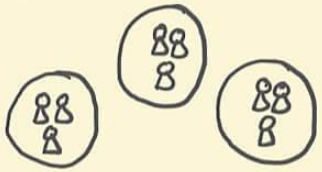
(Source: Adrian Cockcroft, "Velocity and Volume (or Speed Wins)," presentation at FlowCon, San Francisco, CA, November 2013.)

> de plus en plus de changements à moindre coûts, délais et risques

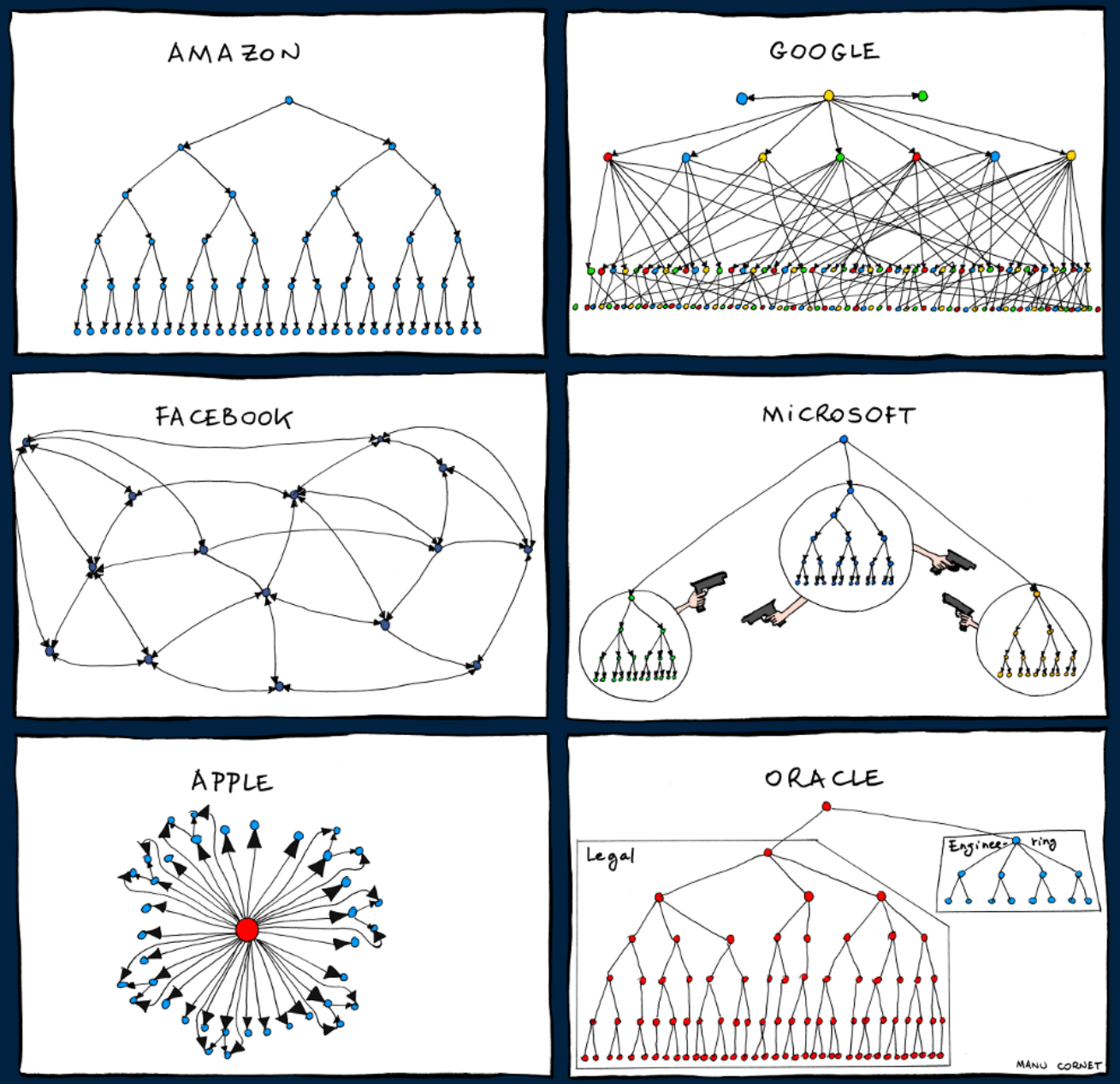
PARAPHRASED

CONWAY'S LAW

THE STRUCTURE OF SOFTWARE WILL MIRROR THE STRUCTURE OF THE ORGANISATION THAT BUILT IT *for example*

ORGANISATION		SOFTWARE
	are more likely to produce	
SMALL DISTRIBUTED TEAMS	→	MODULAR, SERVICE ARCHITECTURE
	→	
LARGE COLOCATED TEAMS		MONOLITHIC ARCHITECTURE

sketchplanations



source(Manu Cornet, www.bonkersworld.net)

?Dev
> personnes qui développent de nouveaux produits, de nouveaux services

?Ops
> personnes responsables de l'exploitation qui est faite de ces produits et services

> The core, chronic, conflict

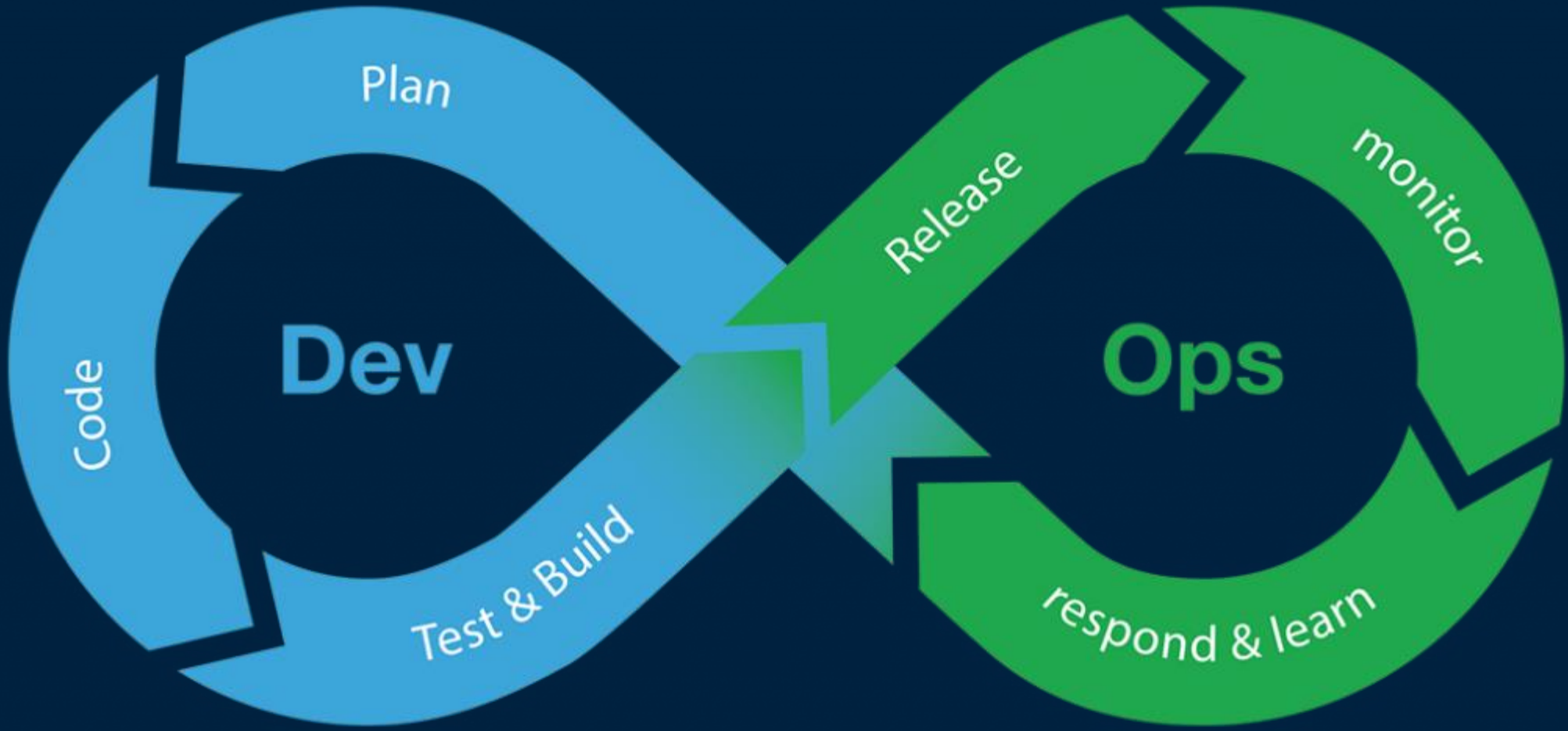
?Dev
> répondre rapidement à des besoins dans un contexte concurrentiel

?Ops
> fournir des environnement stables, fiables, sécurisés


WORKED FINE IN DEV
OPS PROBLEM NOW
memegenerator.net

source(Ward Cunningham, 1992)

> Comment réconcilier les Devs et les Ops ?


Plan
Code
Dev
Test & Build
Release
monitor
Ops
respond & learn

> La promesse DevOps

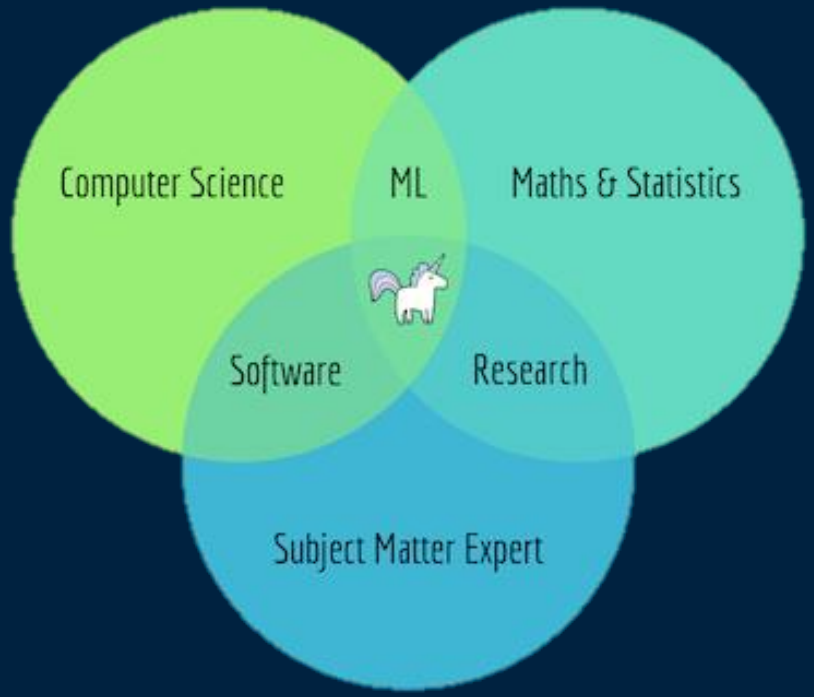
centré sur l'utilisateur
des petites équipes
des boucles de rétrocontrôle régulières
des bugs corrigés tôt
prendre des risques est valorisé
amélioration et apprentissage continu
des **métriques** pour objectiver

```
> quote(”Not all debt is bad, but all
debt needs to be serviced. Technical debt
may be pay down by refactoring code,
improving unit tests, deleting dead code,
reducing dependencies, tightening APIs
and improving documentation”

- Sculley et al. in Hidden technical Debt
in ML Systems)
```

> En quoi ça vous concerne ?
> _

> En quoi ça vous concerne ?
> R est un langage de programmation pour réaliser du développement logiciel

> plot(unicorn)



source: Data scientist Venn diagram, Steven Geringer

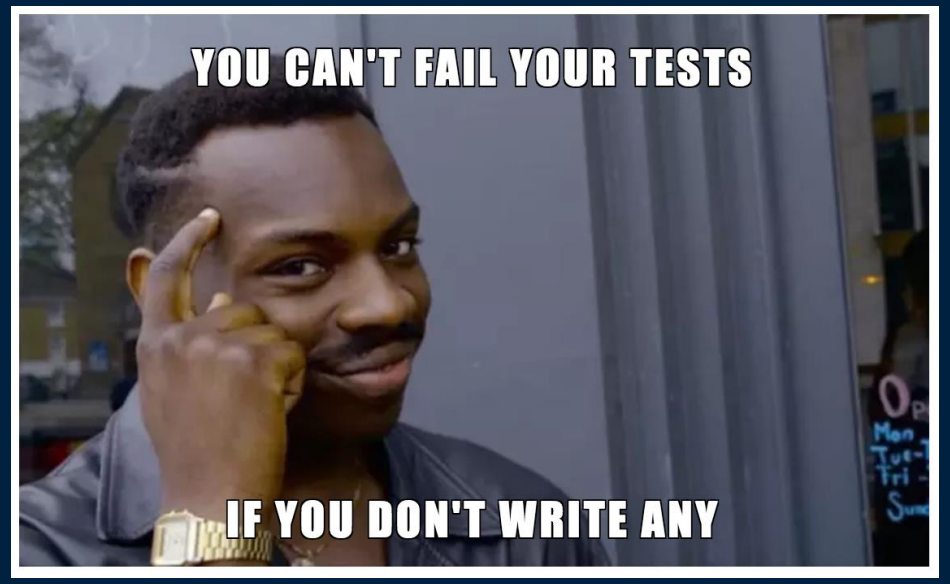
> Comment ?!
> _

```
> Comment ?
> git

# quote(”Le point central c’est
git. git, c’est la vérité !”

- Christophe Dervieux)
```

> Comment ?
> git
> projets RStudio
> .Rmd first

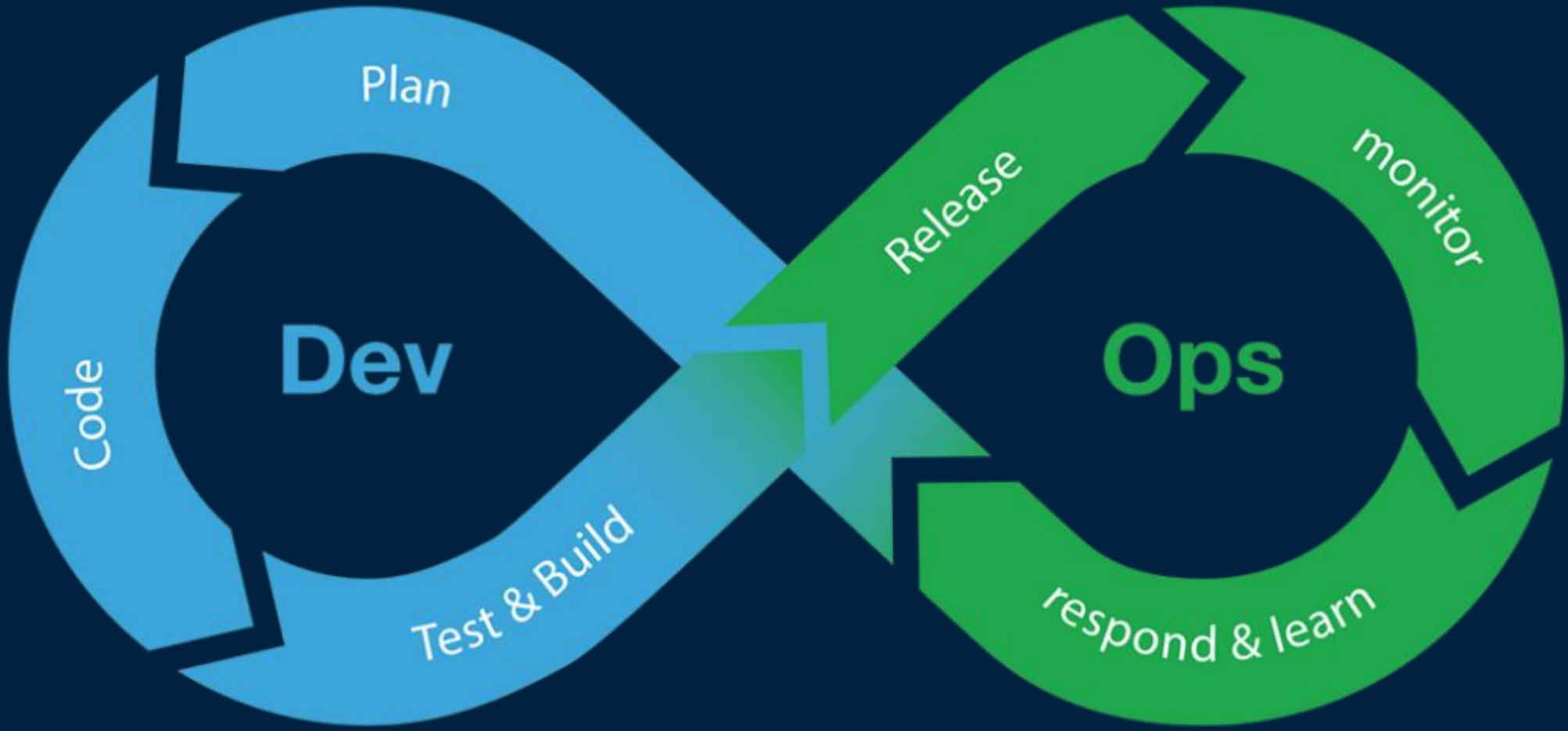
```
# https://rtask.thinkr.fr/fr/quand-le-developpement-commence-par-la-documentation/
```


> Comment ?
> git
> projets RStudio
> .Rmd first
> package first
> tests

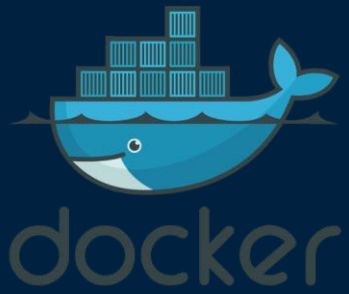



```
# https://thinkr.fr/transformer-plusieurs-scripts-eparpilles-en-beau-package-r/
```

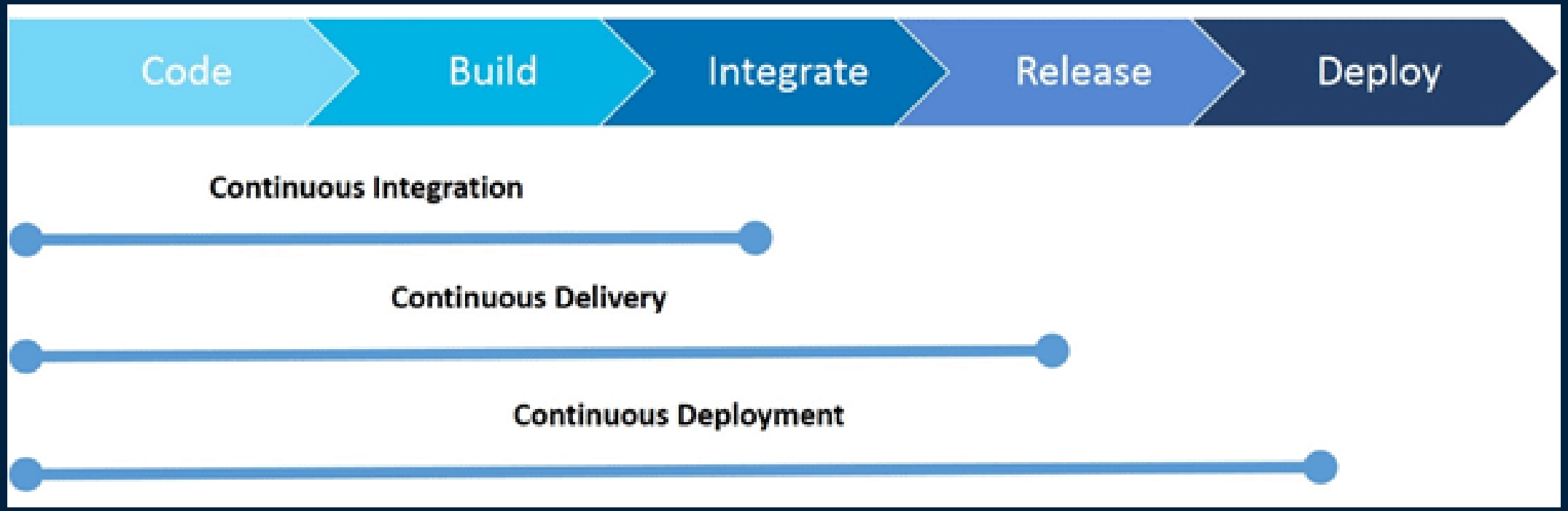

> Comment ?
> git
> projets RStudio
> .Rmd first
> package first
> tests
> intégration continue


Plan
Code
Dev
Test & Build
Release
monitor
Ops
respond & learn

> Comment ?
> git
> projets RStudio
> .Rmd first
> package first
> tests
> intégration continue
> déploiement/livraison continu(e)

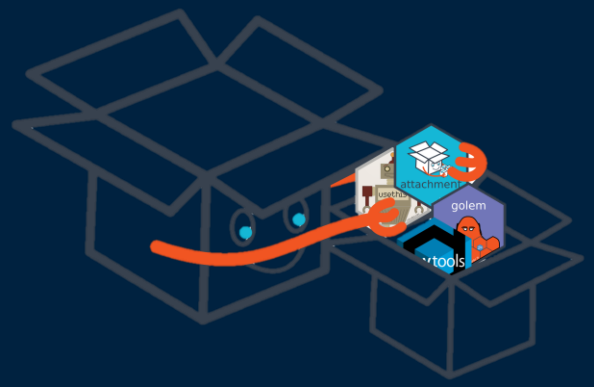



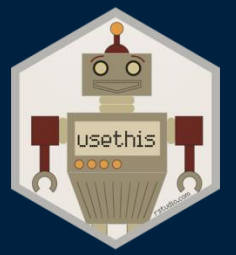
> CI, CD et CD ?


Code
Build
Integrate
Release
Deploy
Continuous Integration
Continuous Delivery
Continuous Deployment

R est parfaitement équipé pour traiter des préoccupations « DevOps » de l'ingénierie logicielle :

> vérifier et évaluer le code fréquemment
> assembler et déployer fréquemment = retours utilisateurs fréquents
> ls("package:development", pattern = "^devOps_" , "^MLOps_", "^statOps_"))





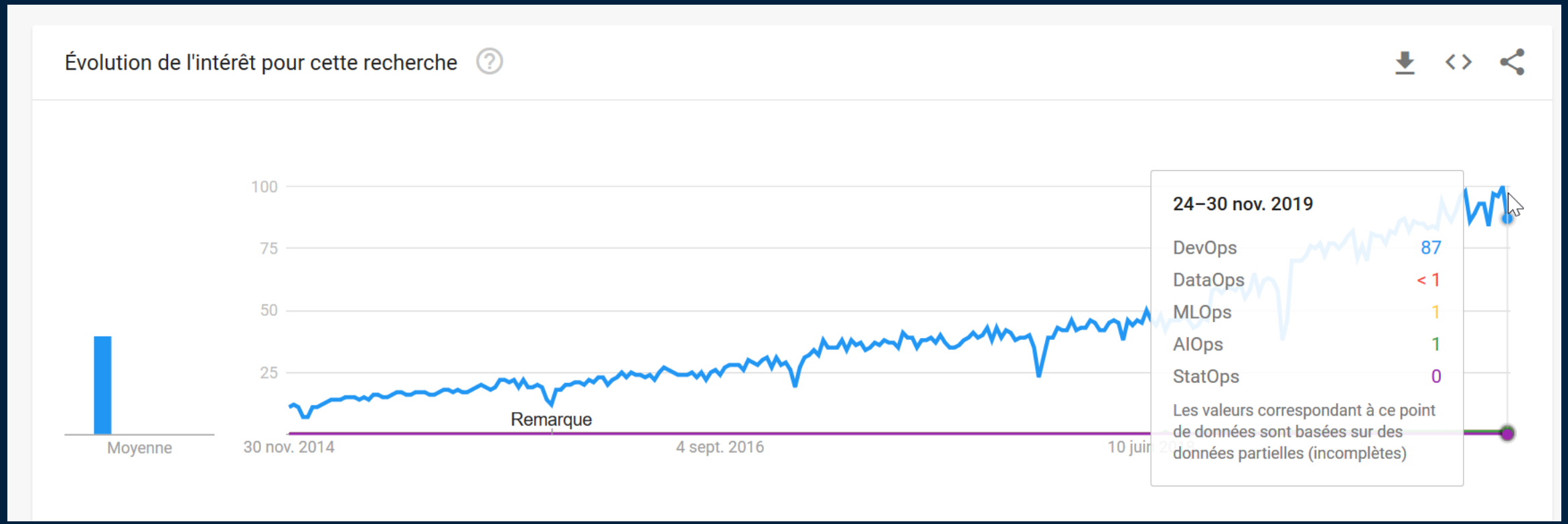
“It is very important to realize that DevOps is not a product. You cannot buy DevOps and install it. DevOps is not just automation or infrastructure as code. DevOps is people following a process enabled by products to deliver value to our end users.”

– Donovan Brown, Principal DevOps Manager @Microsoft

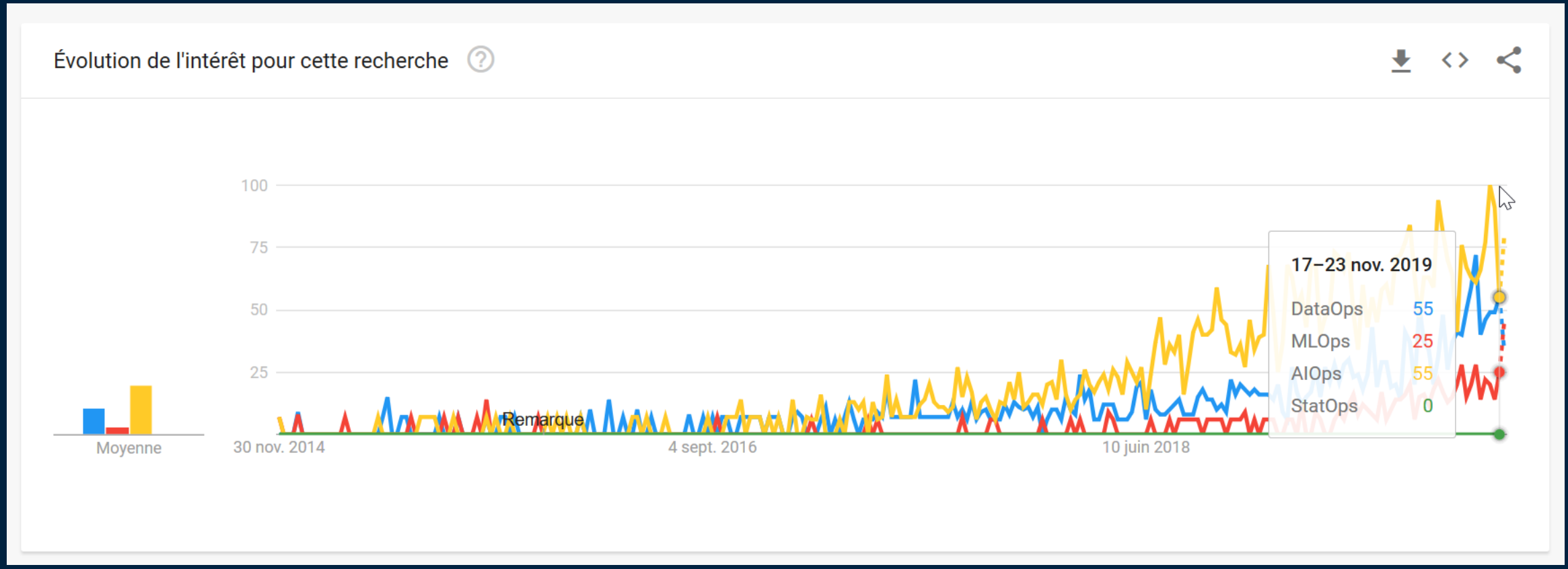
```
> Des process ?

# des bonnes pratiques de programmation R
# des bonnes pratiques de développement
# un git workflow
# un souci constant de « la suite », l'aval
# à partir de 1 personne
```

> Et les déclinaisons [^(?!Dev)]Ops ?

> Et les déclinaisons [^(?!Dev)]Ops ?



> Et les déclinaisons [^(?!Dev)]Ops ?



```
> DevOps + des données
> DataOps

?DataOps
# automatiser la conception, le déploiement et la gestion de la livraison des données
# des process de contrôles statistiques des data
# le data version control
```

```
> DataOps + des maths, des stats, des modèles, des prédictions, des effets…
> (Stat|ML|AI)Ops

?(Stat|ML|AI)Ops
# des process de caching, model tracking, versioning, monitoring et  management
# des questions éthiques
```

?(Stat|ML|AI)Ops

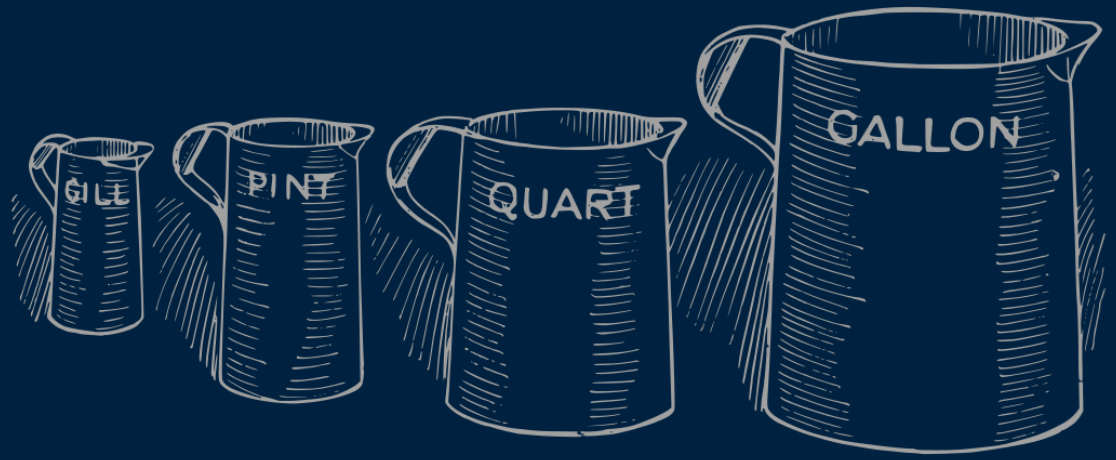
CACE - change anything changes everything (les données en input mais aussi les hyper-paramètres, l'échantillonnage, les seuils…)

les cascades : model_A repose sur model_B qui repose sur un prototype

Abstraction debt : pas de modèle conceptuel unifié

```
?(Stat|ML|AI)Ops
# des métriques à inventer ?
   - temps d'implémentation d'un
nouvel algorithme ?
   - temps pour rendre un
collaborateur opérationnel ?
   - degré de dépendance entre les
modèles ?
```

> plot(foundationaldebt)



> Et vous, c’est quoi votre excuse ?

Merci !
des questions ? (y/n)
>_

> Comment ?

flow,
feedback
continual learning and experimentation

> Comment ?
> Se doter de métriques pour objectiver

Nb de commits/jour
Nb de builds/jour
Nb de test et test case/jour