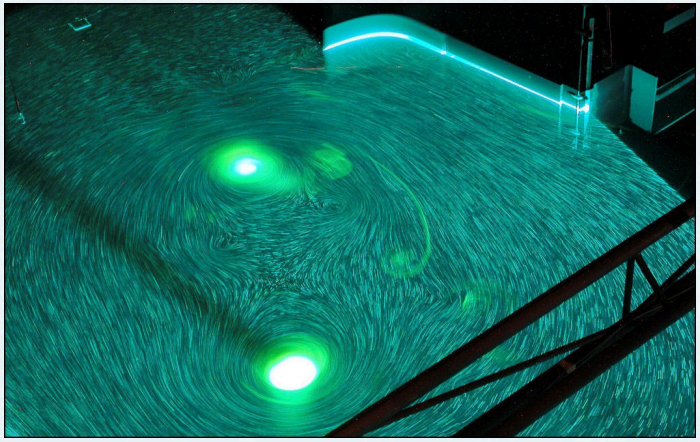
Recherche reproductible
et
Enjeux environnementaux

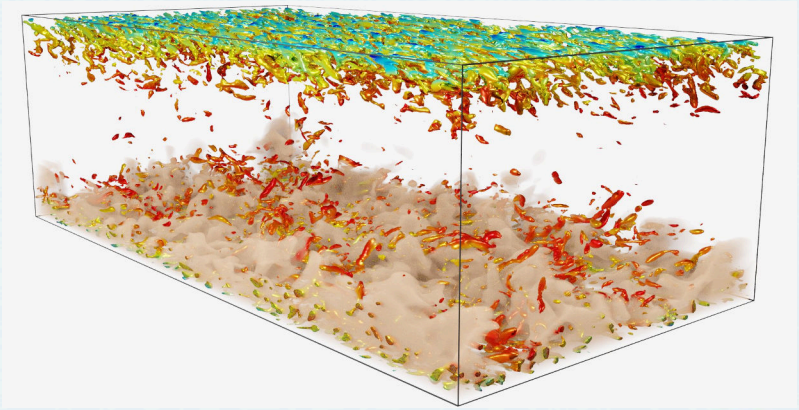
Cyrille Bonamy, Laurent Bourgès

2021.04.06

La recherche reproductible, Qu’est ce que c’est?

C’est reproduire un résultat scientifique, issu d'un workflow complet, comme par exemple une expérience, qu’elle soit concrète (de terrain) ou numérique.





Mais que veut dire “reproduire”?

La recherche reproductible, Qu'est ce que c'est?

Reproduire = obtenir les mêmes "sorties", pour les mêmes "entrées" à chaque étape du workflow (données, paramètres).

Exemples de "sorties" :

- **Grandeurs caractéristiques** (μ,σ) **du phénomène étudié**
- **Champs instantanés ou moyennés**
- **Comparaison au bit près des fichiers de sortie**

Plusieurs déclinaisons de la reproductibilité

Reproductibilité : forte *vs* faible (réplicabilité ?)

Forte

Fichiers de sortie identiques au bit près (checksum)

- env : architecture, random seed
- parfois impossible

Faible

Résultats "similaires / semblables"

- critère de reproductibilité; par exemple : delta < epsilon (ndiff)
- reproductibilité partielle du workflow (code ou données non libres)

- code + données accessibles (versioning)
- protocole : scripts, paramètres

Trouver le bon compromis, et aller vers plus de reproductibilité

La recherche reproductible, Pour qui ? A quoi ça sert ?

Premier bénéficiaire = Vous !

A quoi ça sert ?

- **Meilleure description des traitements effectués / expériences / simulations**
- **Traçabilité, "Recette de cuisine"**
- **Transmission entre équipes / étudiants / collaborateurs**
- **Trois ans après, on peut espérer comprendre ce qu'on avait fait.**

Les enjeux environnementaux dans tout ça

La reproductibilité implique tout un tas de pratiques, et notamment vis à vis des aspects numériques (données et logiciels).

En première approche, on peut donc penser que la Reproductibilité a un coût à court terme (humain, financier mais aussi environnemental).

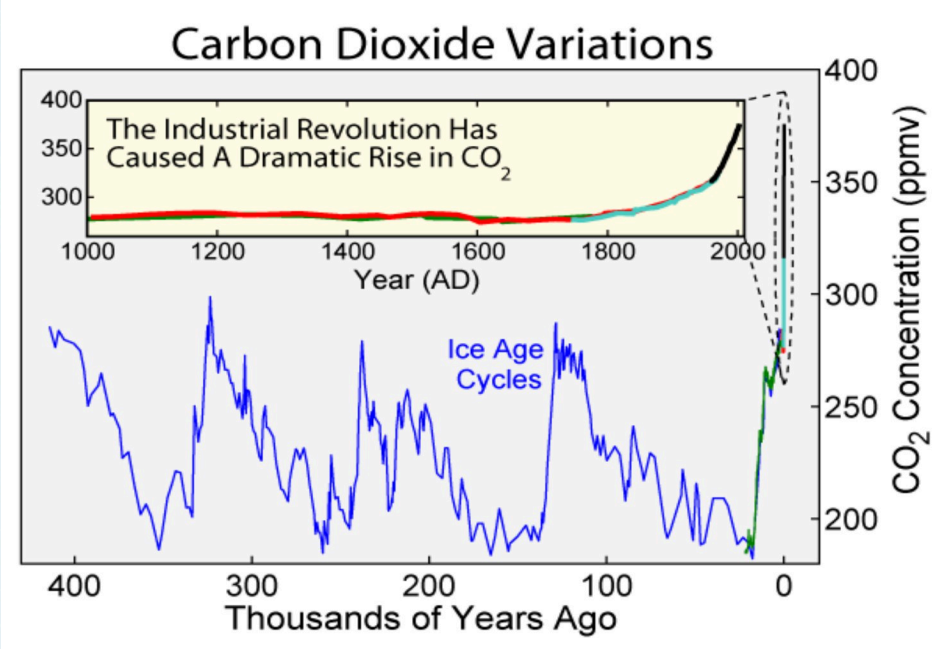
MAIS, sans parler des économies à plus long terme, la reproductibilité n'est elle pas l'essence même de la Recherche ?

On va se focaliser sur les impacts environnementaux des aspects numériques, et essentiellement sur les données et logiciels.

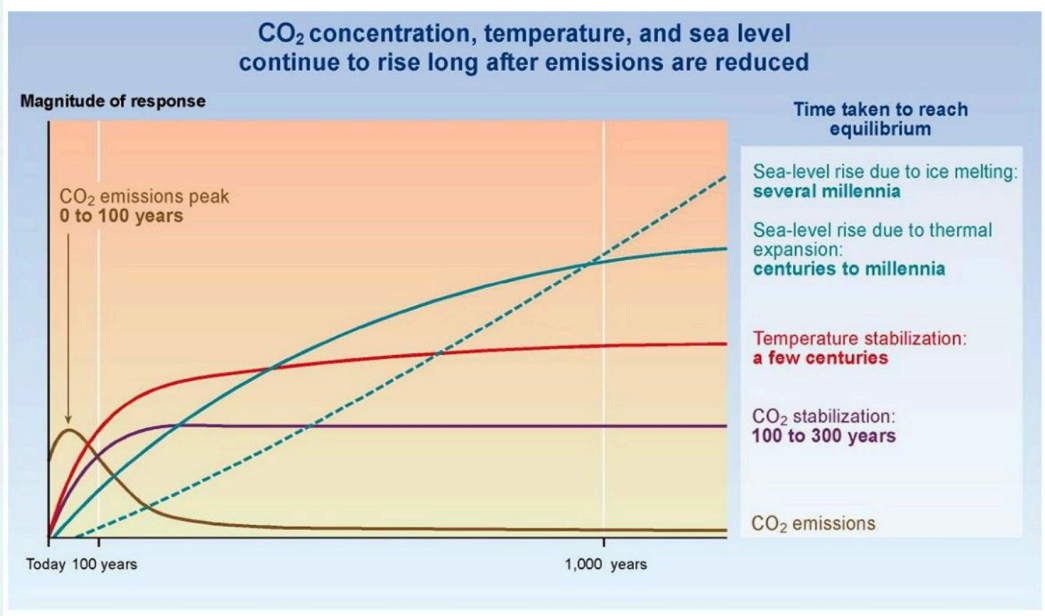
Le GDS EcoInfo

- **Groupement de Service :** des ingénieurs, chercheurs et étudiants à votre service !
- **Agir pour réduire les impacts (négatifs) écologiques et sociétaux des TIC**
- **Quelques exemples de service :**
  - Audit Datacentre (CoC)
  - MatInfo
  - Ecodiag
  - Sensibilisation
  - Formation
  - Veille technologique
- **Recherche**

http://ecoinfo.cnrs.fr

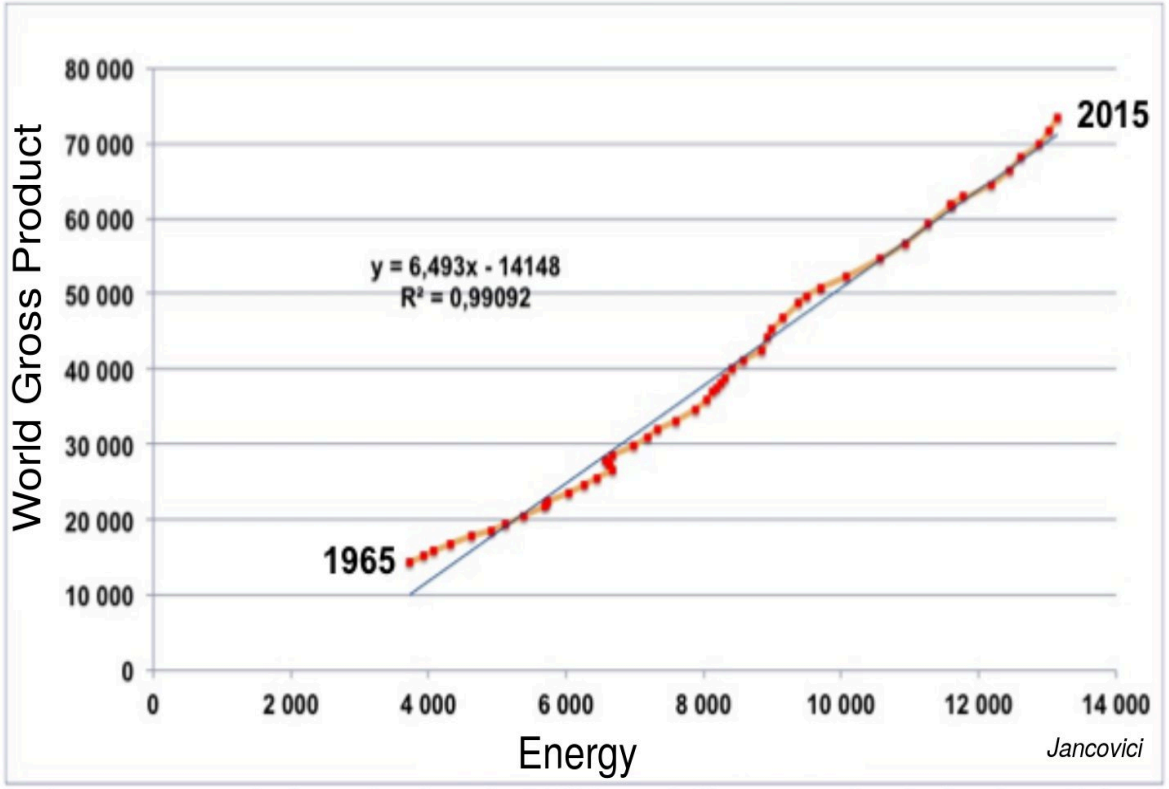


Pourquoi s'intéresser aux impacts environnementaux?

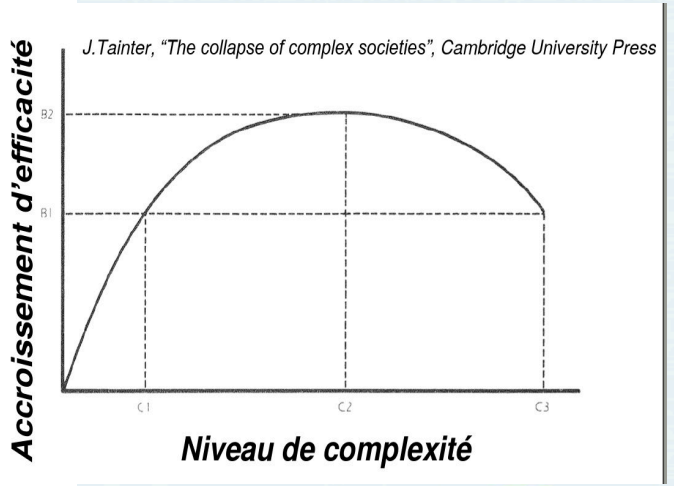


Source: Pierre-Yves Longaretti

Complexité, énergie, croissance économique



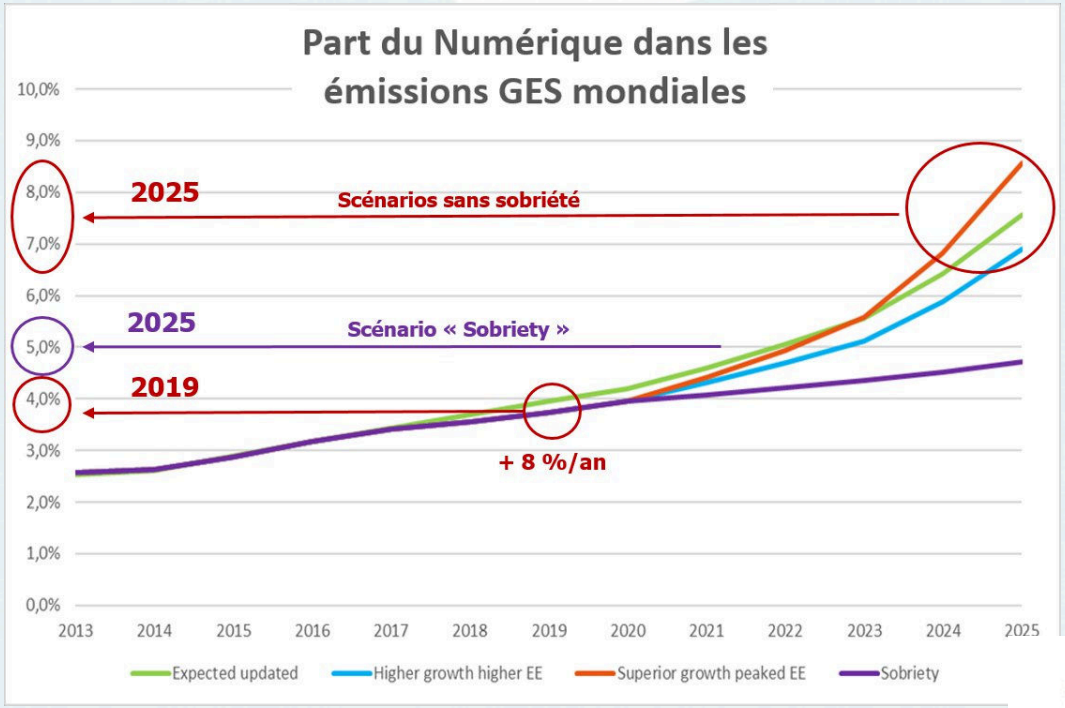
- Croissance = énergie croissante
- Complexité vs efficacité



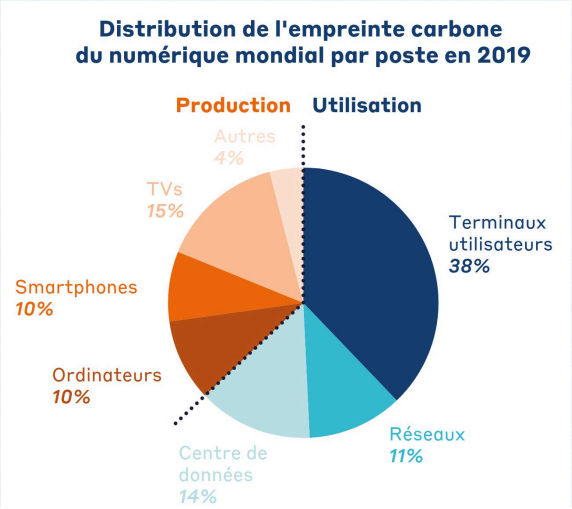
Strict correlation between growth of complexity and of energy use

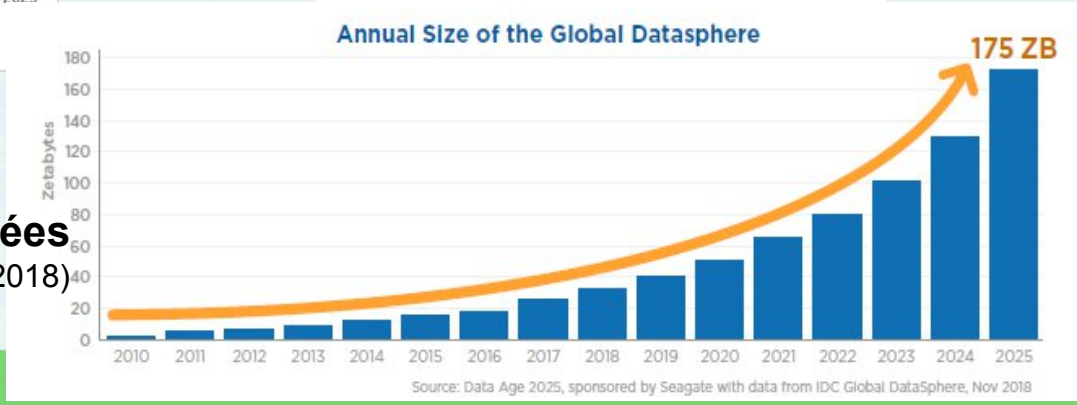
Source: Pierre-Yves Longaretti

Impacts du numérique: croissance exponentielle !



The Shift Project - Déployer la sobriété numérique (2020.10 et 2021.03))

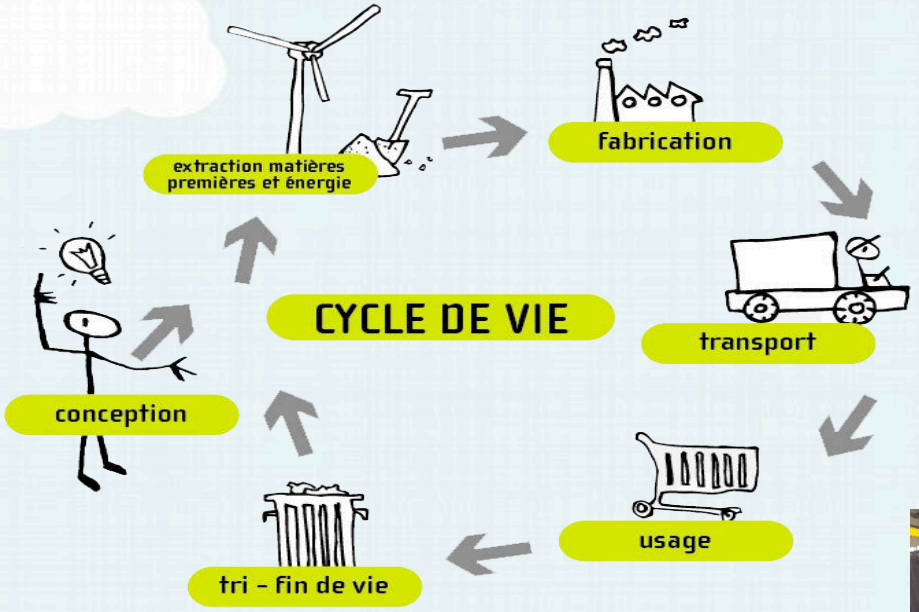




Explosion des données

Source : (Reinsel, D., Gantz, J., Rydning, J., 2018)

Plus concrètement, l'impact du numérique



https://inhabitat.com

https://journalintegration.com/

Numérique: pas ~~virtuel~~, très matériel

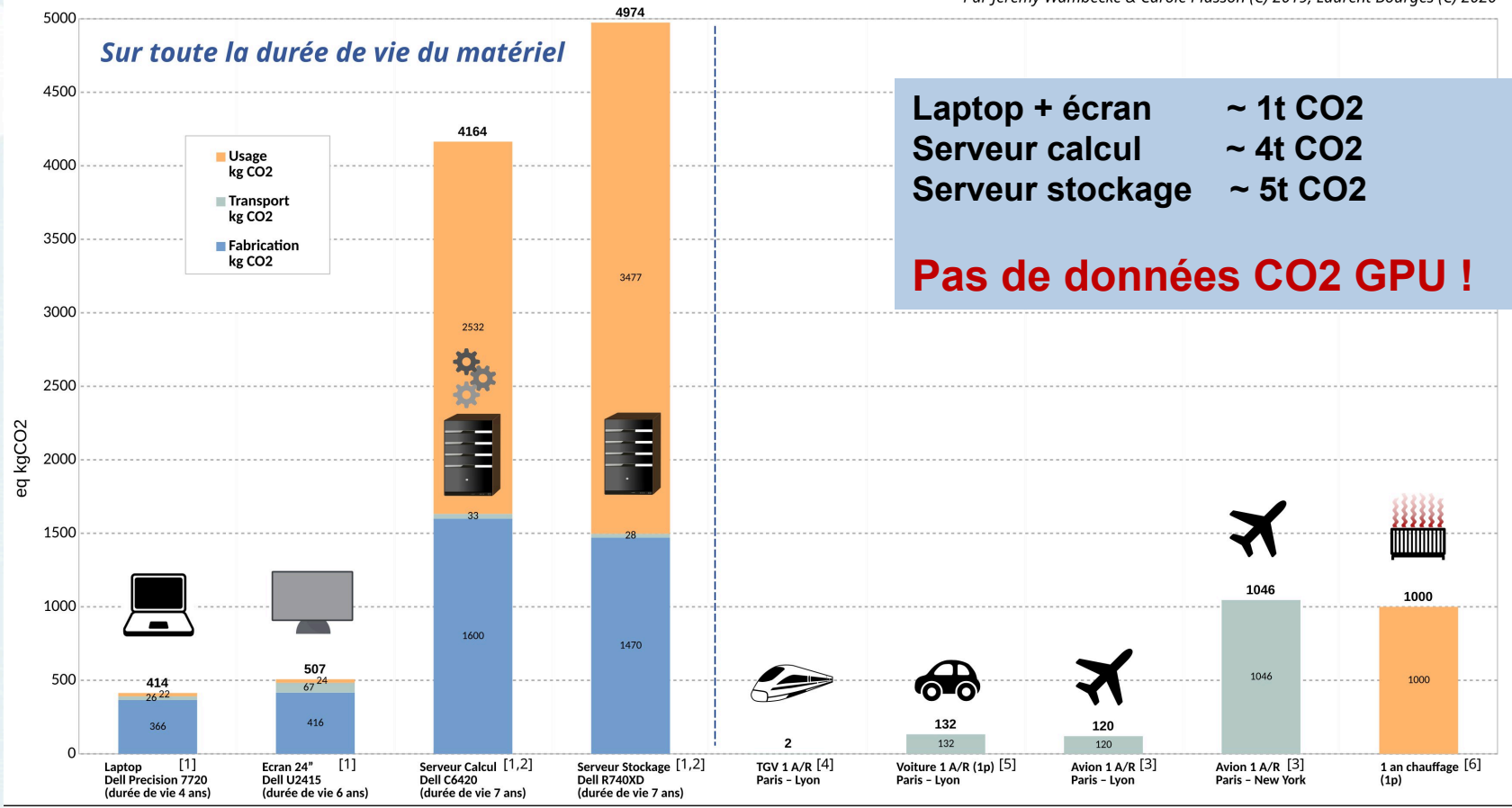
- **Analyse du Cycle de Vie du service numérique (ACV), impacts matériaux, impacts sociaux, obsolescence accélérée**
- **Impacts liés à la durée de vie complète des services, matériels et leur consommation énergétique**

Quelques chiffres des impacts CO2 du numérique

Unité	eq CO2	
	Hardware	
Laptop	350 kg	Fab + Tr (ecodiag)
Serveur	1500 kg	Fab + Tr (ecodiag)
	Usage	
Usage Serveur	2500 kg	Consommation + PUE=1.4 (clim)
Usage laptop + écran (1an)	10 kg	Consommation seule
1h.coeur (HPC)	5 g	Estimation GRICAD
Stockage 1 Go par an (scratch)	15 g	Fab + Tr + Usage
Stockage 1 Go par an (redondant)	35 g	Fab + Tr + Usage
Transfert 1Go (Paris - Orsay)	0.5 g	Etude Renater (stage)
Transfert 1Go (Paris - Montpellier)	1.5 g	Etude Renater (stage)

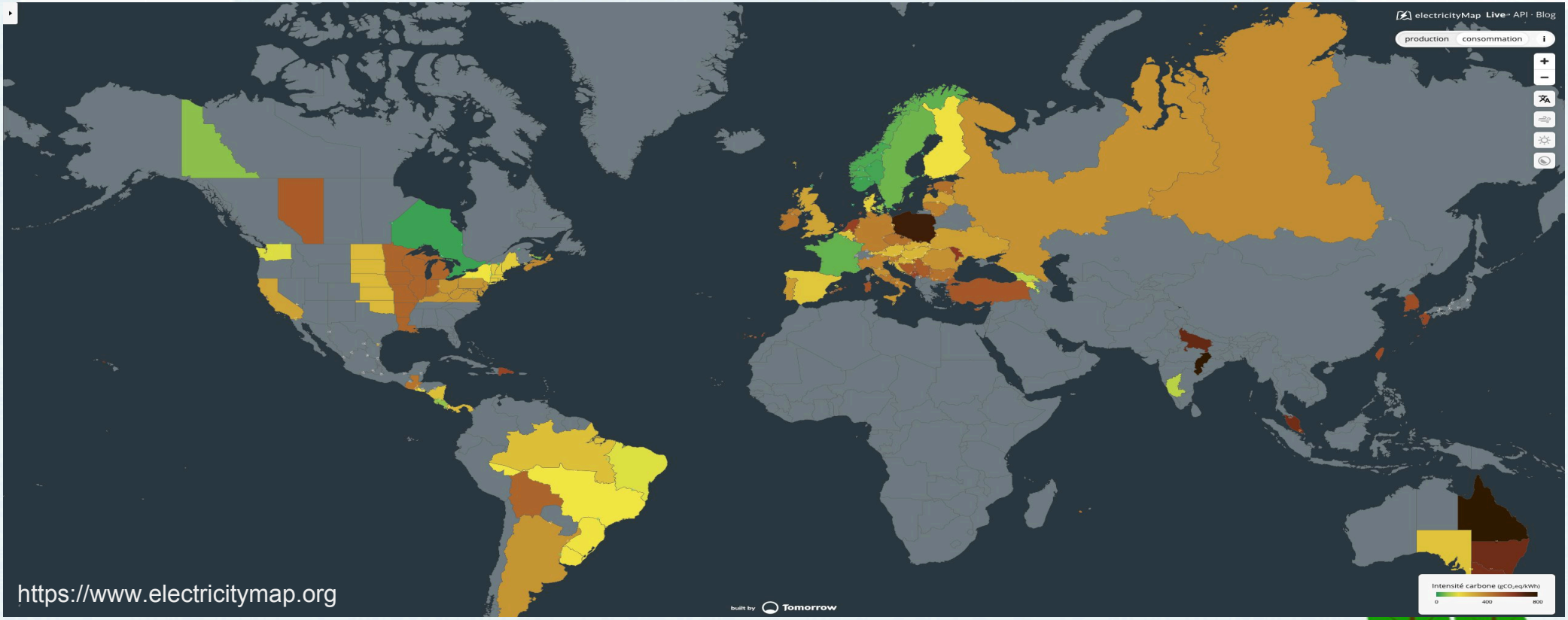
Comparatif d'émission eq CO2

Par Jérémy Wambecke & Carole Plasson (C) 2019, Laurent Bourgès (C) 2020



[1] Données Fiches Dell (usage corrigé pour usage FR) :
(https://www.dell.com/learn/us/en/uscorp1/corp-comm/environment_carbon_footprint_products)
[2] Usage à partir de la consommation moyenne (Berthoud et al. 2020) d'un noeud = 275W (C6420), 375W (R740XD) (https://hal.archives-ouvertes.fr/hal-02549565)
[3] https://eco-calculateur.dta.aviation-civile.gouv.fr/
[4] https://ressources.data.sncf.com/explore/dataset/emission-co2-tgv/table/
[5] Trajet de 473km, pour une voiture émettant 140g CO2/km
[6] https://www.insee.fr/fr/statistiques/fichier/1281320/ip1445.pdf
Facteur d'impact : 0,108 kgCO2e/kWh (FR)

Phase d'usage :
Où sont mes traitements et données ?

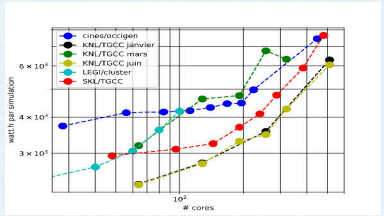


Travaux récents sur les impacts du numérique

- **JRES,** 2019 :

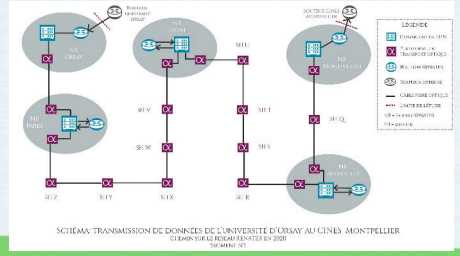
Bonamy, C., Lefèvre, L. & Moreau, G. 2019. **Calcul haute performance et efficacité énergétique : focus sur OpenFOAM.**

Est ce que scalabilité implique scalabilité énergétique?



- Stage **Renater/EcoInfo**, 2020 :

Marion Ficher, Françoise Berthoud, Anne-Laure Ligozat, Patrick Sigonneau. **Évaluation de l'empreinte carbone de la transmission d'1Go de données sur le réseau Renater**



- **JCAD**, 2020 :

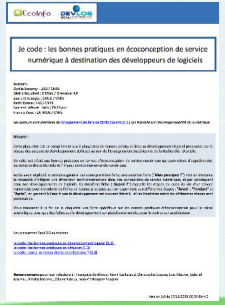
Bonamy, C., Berthoud, F., Bzeznik, B., **Estimation de l'empreinte carbone d'une heure.coeur de calcul**

- 2020 (A venir) :

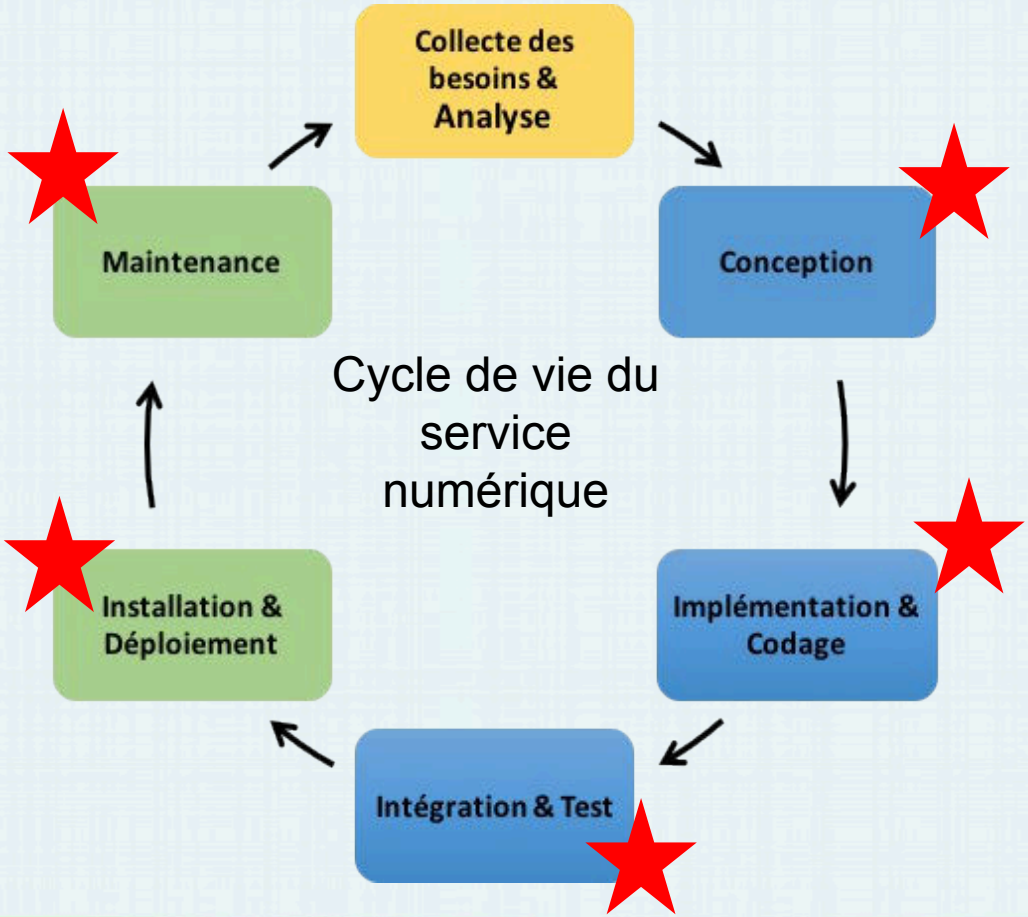
Guillaume Charret, Alexis Arnaud, Françoise Berthoud. **Étude GRICAD de l'empreinte carbone du stockage d'1Go sur un an**

- Guide **EcoInfo**, 2020 :

Bonamy, C., Boudinet, C., Bourgès, L., Dassas, K., Lefevre, L., & Vivat, F. **Je code : les bonnes pratiques en éco-conception de service numérique à destination des développeurs de logiciels**



Quelques bonnes pratiques en développement logiciel



Avant
le développement logiciel

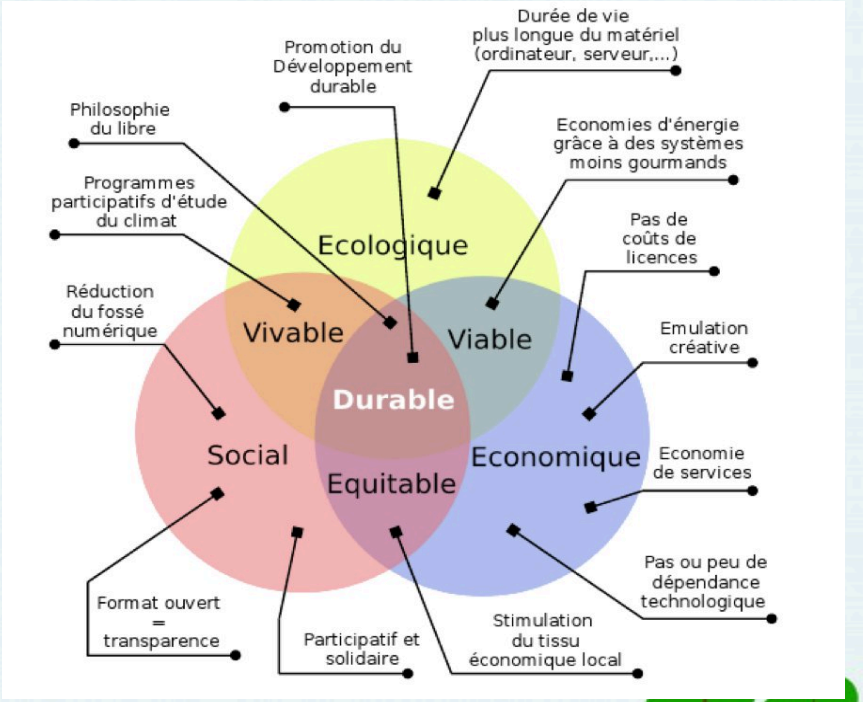
Je réfléchis au nombre de fonctionnalités du logiciel : éviter l'obésiciel

Trop de fonctionnalités disparates : Mon pipeline devient une véritable "usine à gaz" !

Plus de fonctionnalités que nécessaire : mon infrastructure ne suffit plus, je dois en changer !

Fonctionnalités justifiées et suffisantes : léger, efficace et *plus aisément reproductible !*

Je favorise le libre : réutiliser des briques logicielles et contribuer aux communs

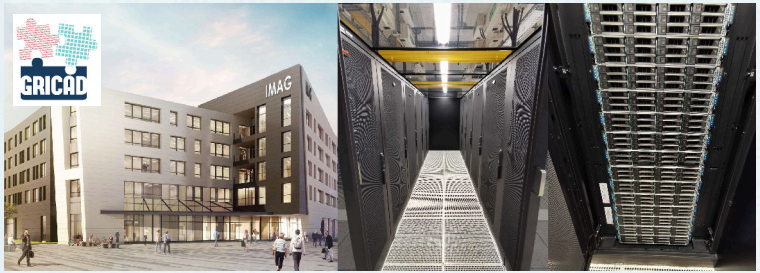


Avant
le développement logiciel

Je réfléchis au déploiement du service : s'adapter au mieux au contexte

En fonction des caractéristiques des plateformes disponibles :

- microcontrôleur ou processeur embarqué
- ordinateur ou serveur local
- plateforme spécialisée : Cluster spécialisé HPC, GPU, FPGA, ARM64
- offre De Service (ODS) de site ou de tutelles (CNRS, Universités)
- clouds publics
- lieu géographique de la plateforme
- capacités disponibles



En fonction des contraintes du service :

- langages supportés, communications spécialisées
- goulots d'étranglement : accès disque, réseau, transferts mémoire
- pérennité, portabilité, sécurité, coûts à long terme, temps de retour
- lieu géographique des usages, tutelles commanditaires du service

Sans oublier les contraintes environnementales …

Avant
le développement logiciel

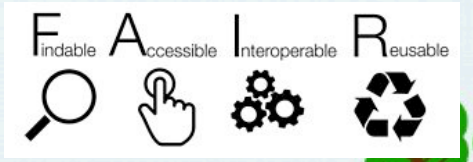
Je planifie la gestion du logiciel : accroître la durée de vie

Un plan de gestion logiciel (SMP Software Management Plan) est un outil pour la pérennisation du logiciel (contexte et caractéristiques du logiciel, organisation de sa diffusion et ses mises à jour)
=> produit modifiable et réutilisable facilement Opidor [5.2], Presoft [5.3]

Je planifie la gestion des données : durabilité, diminution des développements redondants

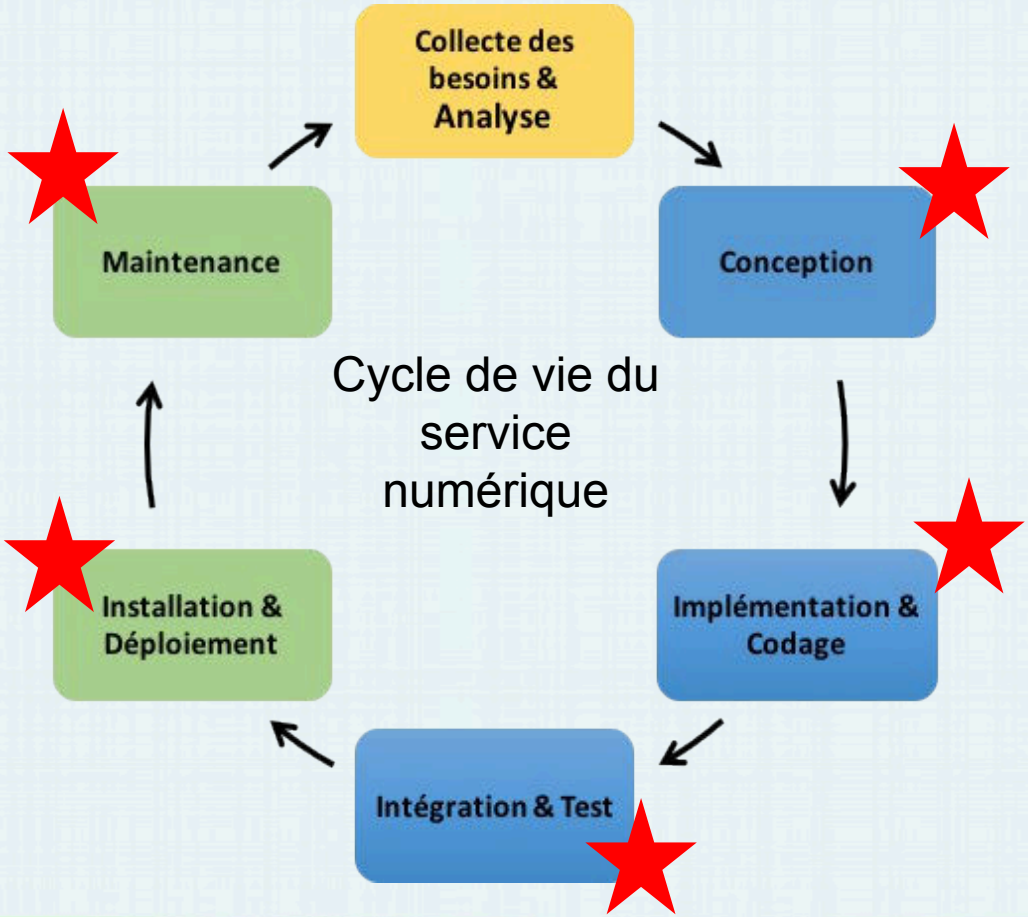
Principes F.A.I.R ("Findable Accessible Interoperable Reusable")
=> diffusion, partage et réutilisation des jeux de données

Plan de gestion de données (DMP) [7.5] : assurer la reproductibilité des données, améliorer l'impact des projets de recherche et leur contribution scientifique => données FAIR.
Exemples : OPIDOR [7.6]



SMP et DMP, des outils facilitant la reproductibilité

Quelques bonnes pratiques en développement logiciel



Pendant
le développement logiciel

Outil de versionning (oui mais…)

J'utilise un outil de gestion de version, mais :

- j'évite ou limite d'y stocker les paquets binaires et les jeux de données non indispensables
- je ne place pas en gestion de version les produits de compilation ni les fichiers de sortie

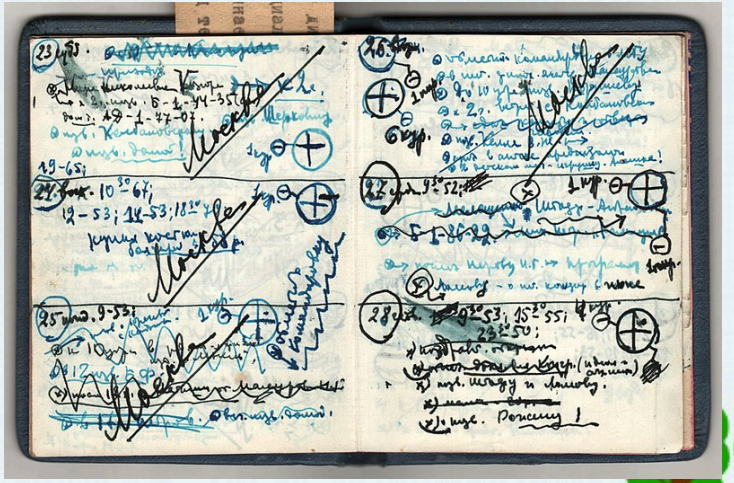
Intégration continue (oui mais…)

- je réfléchis à mon Intégration Continue (CI). Je choisis un docker de taille minimum, j'active ma CI uniquement sur certaines branches et j'envisage une exécution programmée.
  Ainsi je n'exécute pas tous les tests et ne produis pas tous les fichiers à chaque modification
- je surveille la durée des jobs, leur nombre, la taille des artefacts, le trafic réseau
- je privilégie les forges mutualisées

Documentation

Je documente mon code :

- pour les utilisateurs, mais aussi pour les développeurs
- je n'hésite pas à utiliser des outils permettant de mixer code et documentation (notebooks)



Boris Parygin Notebook spread/ 1966

Pendant
le développement logiciel

Je m'impose des normes de codage et prévois des tests (oui car...)

- Meilleure lisibilité
- Éviter les régressions
- Ré-usage

Markus Spiske, wikimedia.org

J'optimise mon code (oui mais...)

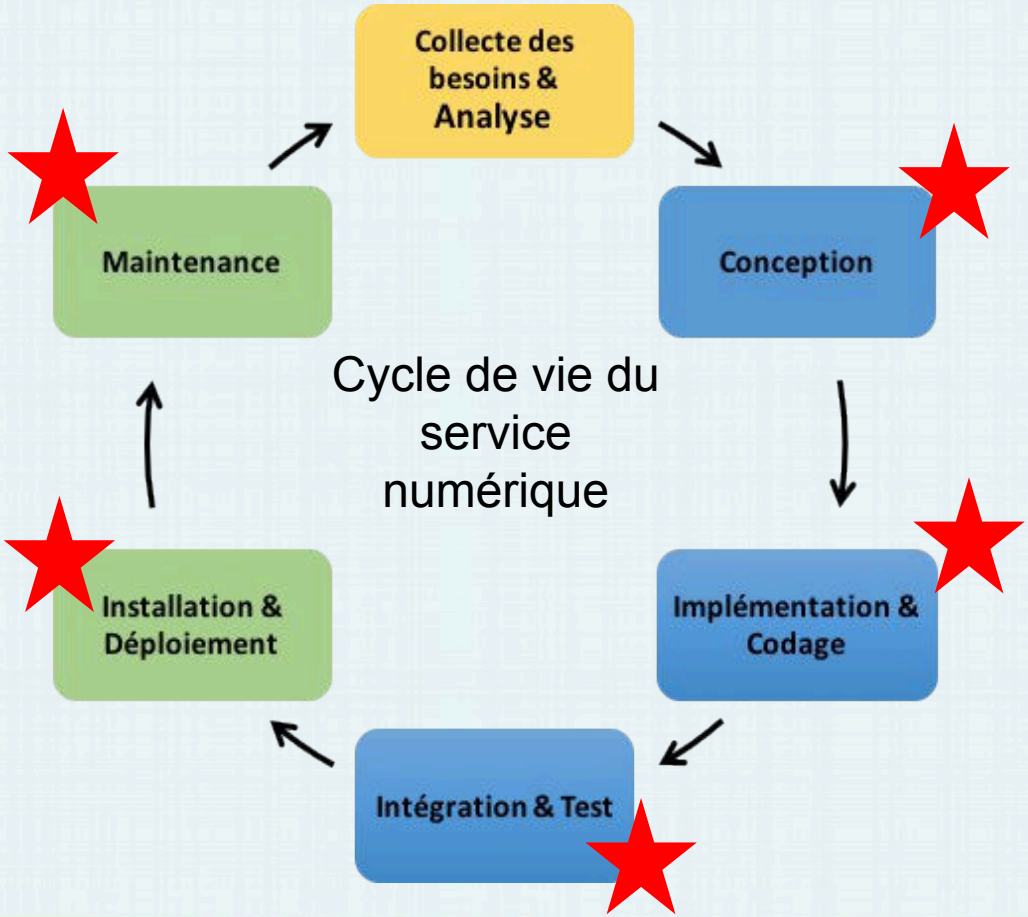
Attention à l'effet rebond :

Optimiser un logiciel peut induire à lancer davantage d'opérations ou traiter davantage de données, donc l'empreinte écologique du service ne sera pas réduite (Paradoxe de Jevons).

L'optimisation devrait servir simplement à réduire la consommation énergétique et des ressources, et si possible d'arriver plus vite au résultat. Chaque exécution a un impact !

Il est primordial de n'optimiser que ce qui a le plus d'impact (Loi de Pareto).

Quelques bonnes pratiques en développement logiciel



Après
le développement logiciel

Déploiement : sobriété numérique

- hébergement mutualisé, labellisé COC,
  au plus près des données et des utilisateurs
- virtualisation,
  sauf cas particuliers (HPC)
- attention aux effets rebond :
  multiplication des machines virtuelles, services

Production : amélioration continue

- supervision et alertes :
  pics CPU, ressources, consommation électrique
- adapter service en fonction des usages
- Réduire fréquences et volumes des sauvegardes

Exemples d'outils de supervision : top, vmstat, zabbix, scalasca, nagios, prometheus, grafana

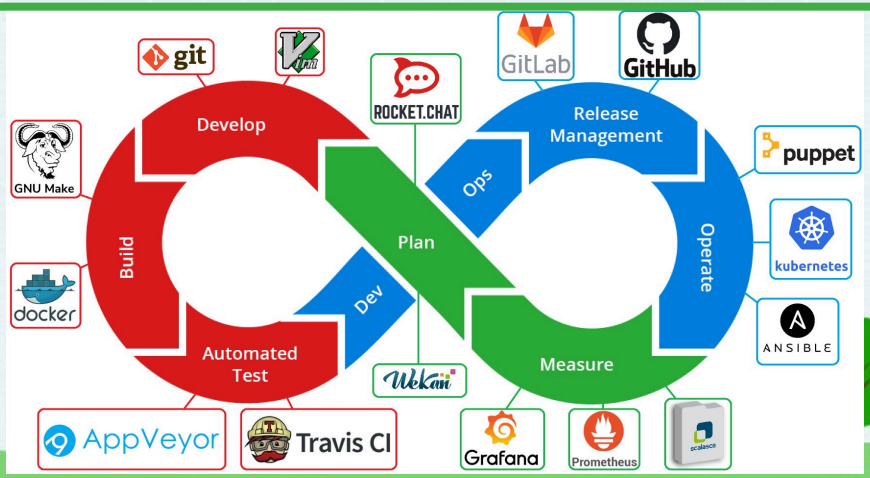
Je distribue et maintiens mon code : favoriser la durabilité et la simplicité

Diffusion

- je dépose le logiciel en un endroit unique et facilement accessible
- déclaration auprès de Software Heritage [11.2]

Gestion des mises à jour

- je réduis la taille des produits logiciels
- je rationalise leur nombre et leur fréquence



Exemples d'outils utilisés pour l'amélioration continue du service numérique
(source : PNGEgg, adaptée par C. Bonamy)

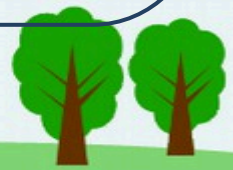
Dans le monde réel ?
Pour commencer, le minimum vital pour plus de reproductibilité

Logiciel

- Versioning des codes et scripts (sans oublier pre et post processing)
- Simplicité, durabilité, norme de codage
- Éviter les obésiciels
- Sensibiliser
- Science ouverte (Open Source)

Données

- Principes FAIR : méta-données (traçabilité, curation), formats standardisés et binaires, durée de vie
- Archivage / Volumétrie : savoir supprimer
- Sensibiliser
- Science ouverte (Open Data)

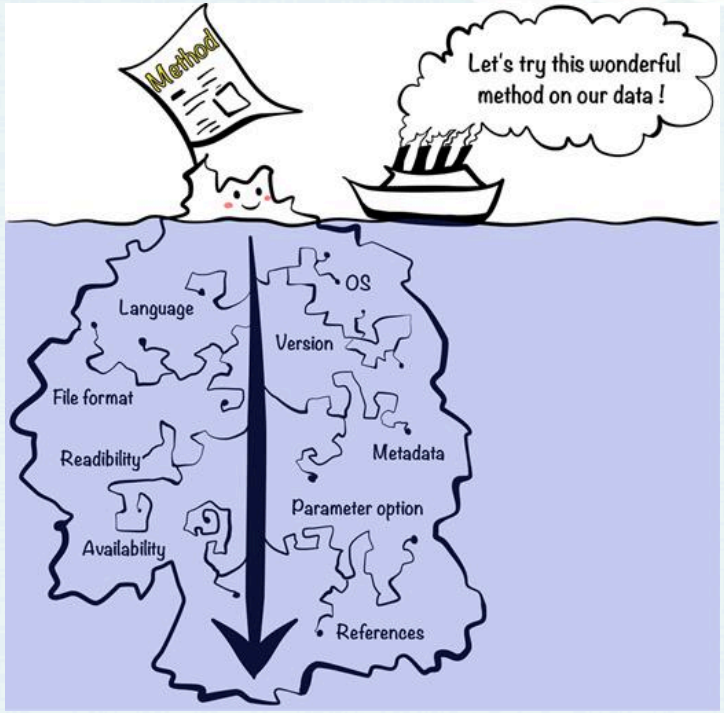


Conclusion générale

La Reproductibilité : une nécessité, pas assez valorisée !

Mais essayons de ne pas oublier les impacts environnementaux de nos activités.

Tout est question de compromis...



https://doi.org/10.1093/gigascience/giy077

Cependant ...

Getty Images – AlexandrBognat / Netflix

Concernant le numérique :

- Impact recherche << numérique mondial
- Impacts IOT / IA / data mining ?

Plus généralement, dans le milieu de la recherche :

- Impact sur l'environnement dominé par les missions et trajets des personnels (> 60% bilan GES ISTERRE 2017)

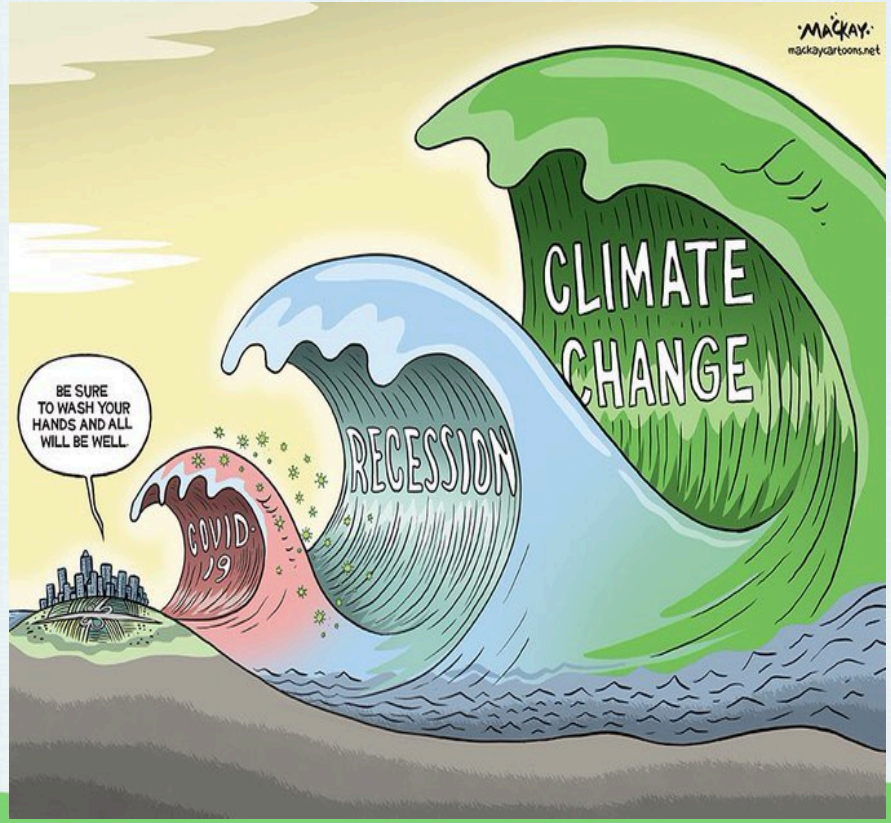
Mais le monde la recherche peut être un levier du changement (exemple : Matinfo)

L'environnement... un problème pour ~~demain~~ **aujourd'hui**

Optimiser n'est ~~pas~~ plus une solution...
Il est bien trop tard !

Il faut faire moins, mais mieux.

<u>Et surtout changer !</u>



Questions ouvertes ?

Sobriété = consommer le juste nécessaire / ne pas gaspiller : achats matériels, heures de calcul, mémoire, stockage, archivage

Politique de gestion des données et des codes :
- Granularité ? garder / archiver tous les data sets, runs, paquets ?
- Pérennité des jeux de données (DOI, zenodo, opidor ?)
- Accumulation (code, data, infrastructures) sur le long terme ?

Problématique liée à la mutualisation :
- Datacentre local ou national ?
- Cloud public ou privé ?

Problématique des budgets et allocations (et donc de la pérennité).

MERCI POUR VOTRE ATTENTION