

PROGRAMMEZ

Le magazine des développeurs

05/
2020

N°240
23e année

Python 3.8

Les nouveautés



Informatique quantique

Amusons-nous
avec les
Qubits !

React Native vs Flutter

Les vrais arguments

Test or not test

Se poser
les bonnes
questions

Télétravail

Bonnes pratiques
Témoignages

Microsoft

Premiers pas
avec
.Net Core IoT

M 04319 - 240 - F: 6,50 € - RD



Le seul magazine écrit par et pour les développeurs

N°3
Sortie :
fin mai

Retour vers le passé :

redécouvrez les ordinaures et les technologies des années 1970 à 2000 !



Commandez directement sur programmez.com

6,66 € (+frais de port*) **36 pages**

Revue trimestrielle. Editée par Nefer-IT. *Avec frais de port : 7,66 €

NOUVEAU ! Abonnement 1 an : **30 €**


```
this.AutoSize = true;
this.Text = "le pourquoi du comment";
```

// ne pas oublier de fêter nos 22 ans

Nous avons réussi après moult péripéties à sortir le 239, même si la version papier subissait un retard d'une dizaine de jours, mais impossible de faire mieux, vu la situation tendue à partir de mi-mars.

Ces contraintes se sont accentuées les semaines suivantes. Et beaucoup de questions se sont posées à nous en préparant les 240 et 241 de Programmez!. La décision ne fut pas facile. Nous sommes conscients que ce n'est pas l'idéal. Mais, nous devons agir pour préserver le magazine. Nous réduisons temporairement la pagination, passant de 84 à 68 pages.

Plusieurs impératifs se sont imposés à nous :

- Économique : avec la mise à l'arrêt, ou quasiment, des transports, les fermetures de kiosques par centaines, le ralentissement des dépenses marketings, nous devons optimiser nos budgets de fonctionnement. L'impression d'un numéro représente +50 % du budget mensuel de Programmez!, ce qui est considérable.
- S'organiser dans la désorganisation : une partie du mois de mars a largement désorganisé notre écosystème. Oui, non, oui, finalement non mais finalement oui. Nous devons réorganiser les plannings de fabrication du magazine jusqu'à 4-5 fois par jour. Heureusement, une stabilisation s'est faite dès fin mars, ce qui nous a permis de relancer la rédaction mais nous sommes fortement impactés par le confinement et le fonctionnement, parfois très ralenti de La Poste et de nos prestataires.
- Répartir les contenus et s'adapter aux contraintes de nos contributeurs techniques : sans nos contributeurs, il n'y aurait pas de magazine. Réduire la pagination, c'est aussi réduire nos relances « Tu en es où dans ton article ? ».

Cette pagination de crise sera reconduite au moins jusqu'au 241.

Évènement un peu anecdotique. Programmez! a fêté en avril son 22e anniversaire ! Désormais, nous sommes dans notre 23e année... Bon anniversaire.

Plus que jamais, votre soutien est nécessaire. Un magazine c'est aussi une délicate équitation financière.

Nous remercions tous nos contributeurs passés et actuels, sans qui, il n'y aura rien à lire. Merci aussi à toute l'équipe : Frédéric, Pierre, Olivier, Otto, Terry, Hindley, Eric.

François Tonic - ftonic@programmez.com

Pigiste dès le n°3 (1998-99)

Rédacteur puis rédacteur en chef adjoint (1999)

// Rédacteur ici et là, pigiste IT, suite & fin de mes études à la Sorbonne

// Création de Toutankhamon Magazine (été 2001)

Rédacteur en chef (depuis 2002)

Rachat de Programmez!, fondateur de Nefer-IT,

lancement de Technosaures (2013- ?)

Brèves	4
Agenda	6
Roadmap 2020	8
Matériel	9
Télétravail	12
Chronique : OS/2	16
Eco-conception web	21

Dossier : tests logiciels 23

Abonnez-vous ! 42

Boutique Programmez! 43

NIVEAU 100



Python 3.8 :
les principales
nouveauités39



InspectIT.....45



React Native vs
Flutter.....48

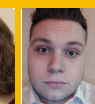
NIVEAU 200



Informatique
quantique
partie 2.....51



Linux : eBPF59



Kubernetes sous Linux
partie 156



.Net Core IoT62

Le développeur est un animal bizarre.....66

MESSAGE PRIORITAIRE :

**Programmez! n°241, prochain numéro
date stellaire : 5 juin 2020**

**Développeur freelance : comment le devenir ?
comment le rester ?**

Dossier spécial Red Hat Openshift



Zoom : grandeur et décadence en temps de confinement

Le grand gagnant de la crise sanitaire actuelle et du confinement, c'est probablement Zoom : cette société américaine propose un outil de visioconférence calibré pour des meetings rassemblant de nombreux utilisateurs. Depuis le début de l'année, l'application est en effet passée de 10 millions d'utilisateurs quotidiens à 200 millions en mars. Mais ce succès s'est aussi accompagné de critiques nombreuses sur la sécurité et la protection de la vie privée des utilisateurs, conduisant le procureur de l'état de New York à ouvrir une enquête officielle sur les pratiques de Zoom en la matière. Après avoir essuyé un tir de barrage de critiques à la fin du mois de mars, la société a annoncé avoir stoppé le développement de nouvelles fonctionnalités et que l'ensemble de ses ingénieurs travailleraient dorénavant sur l'amélioration de la sécurité et de la protection de la vie privée.

NDLR : pour Zoom, il faut agir vigoureusement car les entreprises interdisent parfois d'utiliser cet outil pour éviter tout problème de sécurité...

StopCovid : le « contact tracing » à la rescousse

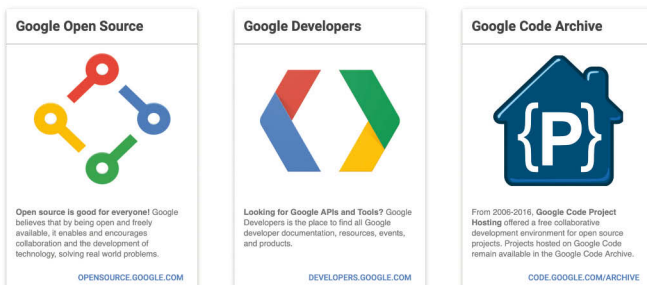
Pour faire face à la pandémie, le gouvernement souhaite développer une application de « contact tracing » qui permettra de retracer les contacts avec d'autres utilisateurs et éventuellement être informé si l'un d'entre eux a été contaminé par le virus. Le genre de technologies qui inquiète évidemment, mais le gouvernement met les bouchées doubles pour rassurer, en promettant un dispositif basé sur le volontariat uniquement, s'appuyant sur le Bluetooth et respectueux du RGPD. Le développement de l'application a été confié à l'INRIA, qui travaille à l'élaboration de ce type de technologies dans le cadre du projet européen Pan European Privacy Preserving Proximity Tracking (PEPP-PT).

Ransomware : des promesses, rien que des promesses

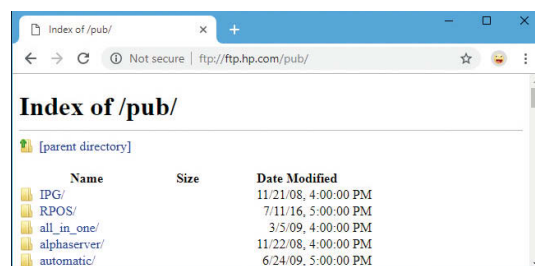
Au début du mois de mars, plusieurs groupes cybercriminels spécialisés dans les ransomwares avaient fait savoir à la presse que pendant la durée de l'épidémie, les hôpitaux et établissements de santé seraient épargnés. Une annonce qui avait fait miroiter à certains l'espoir d'une trêve bienvenue, mais qui s'est révélée assez décevante : on apprenait quelques jours plus tard qu'un hôpital de Prague était paralysé par une attaque de ransomware, et l'un des groupes à l'origine des déclarations annonçait sur son propre site avoir attaqué une société pharmaceutique britannique. Ils assurent néanmoins que depuis, ils n'en ont pas attaqué d'autres. Au début du mois d'avril, Interpol a indiqué dans une alerte avoir détecté une hausse des

Google Code Search : Google prête ses jouets

Google a ouvert aux internautes l'un de ses outils jusque-là limités à ses employés : Google Code Search, un outil qui permet aux utilisateurs de rechercher des termes spécifiques dans le code source de plusieurs projets Open Source maintenus par Google. Il est ainsi possible de rechercher une fonction spécifique au sein de projets Google et de comprendre comment celle-ci est utilisée au sein des différents projets. L'outil dispose de plusieurs filtres qui permettent de limiter la recherche à certains langages ou à certains projets, afin de simplifier la tâche des utilisateurs.



Firefox, Chrome : Adieu FTP



Chrome et Firefox ont annoncé, à quelques semaines d'écart, leur intention de mettre fin au support du protocole FTP utilisé pour transférer des fichiers via des navigateurs web. Trop daté, difficile à maintenir et peu sécurisé, le protocole semble aujourd'hui voué à une mort prochaine. Chrome a déjà commencé à couper progressivement le support depuis sa version 80 et Firefox prévoit d'amorcer le processus à partir de sa version 71, avant de couper complètement le support en début d'année 2021.

tentatives d'attaques visant les établissements de santé. Les espoirs de trêve ont fait long feu.

Apple bloque les cookies tiers dans Safari

Avec la version 13.1 de son navigateur Safari, Apple a officialisé le blocage de l'ensemble des cookies tiers sur son navigateur. Cela signifie que les seuls cookies acceptés seront issus du site

consulté par l'utilisateur, et plus ceux de la myriade de partenaires publicitaires qui gravitent autour. Une évolution qui est avant tout symbolique : les évolutions récentes des navigateurs rendent la vie difficile pour les cookies publicitaires et celui-ci n'est plus l'outil de choix de la publicité en ligne. Heureusement pour les annonceurs, il leur reste tout un arsenal de techniques en réserve pour pister les internautes.



**Lunelotte et la
réciproque mystérieuse**

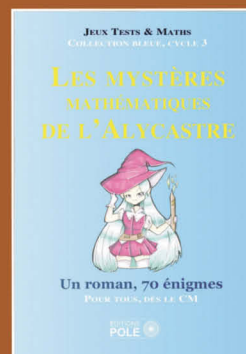
EDITIONS
POLE



Les nouveautés

**Une BD pour
découvrir la logique**
Tous publics
Lycéens
dès la seconde

**Un roman
à énigmes**
70 problèmes
à résoudre
dès le CM1



**Les mystères mathématiques
de l'Alycastre**

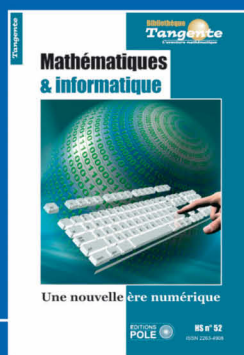
Disponibles chez votre libraire ou auprès des Éditions POLE
à l'adresse <https://infinimath.com/librairie>



**Intelligence
artificielle**



Les graphes



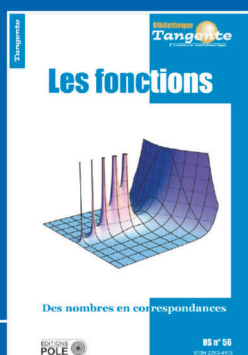
**Mathématiques et
informatique**



Les nombres



Les algorithmes

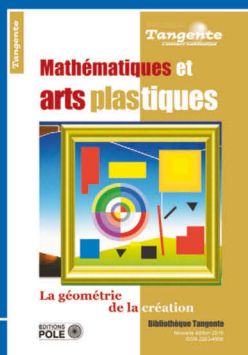


Les fonctions

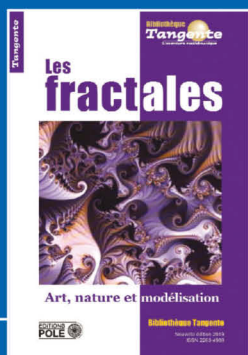
Bibliothèque **Tangente** L'aventure mathématique

**La plus belle collection
mathématique au monde**
(déjà 70 ouvrages en couleur de 160 pages)

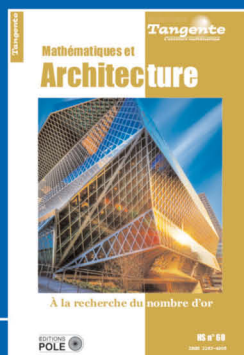
**Pour les recevoir, rendez-vous sur
<https://infinimath.com/librairie>
ou abonnez-vous SUPERPLUS à
Tangente et recevez-en 4 par an**



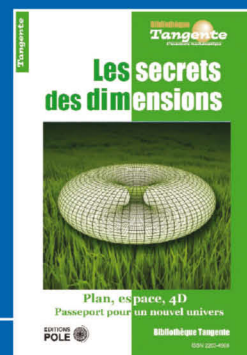
**Mathématiques et
arts plastiques**



Les fractales



**Mathématiques et
architecture**



**Les secrets des
dimensions**

Avertissement : avec le Coronavirus et les restrictions ordonnées par le gouvernement, de nombreux évènements et conférences pour les développeurs ont été annulés et reportés. Nous faisons un point sur la situation connue au moment du bouclage de Programmez! Nous mettons régulièrement à jour sur www.programmez.com

27 & 28 mai : Big Data Paris

Salon annulé.

Juin

12 : DevFest Lille/Lille

Edition annulée.

Retour le 11 juin 2021.

Juillet

1er : DevOxx France/Paris

Annulation de l'événement.

Objectif : 2021

Août

13-16 : KubeCon / Amsterdam

La grande conférence européenne sur Kubernetes est reportée à juillet ou août. Pas de nouvelles connues.

Septembre

1er : JFTL / Montrouge

La journée des tests logiciels revient pour la 12e année. Le but est de montrer, expliquer les tests, et pourquoi ils sont si importants. 1 000 personnes sont attendues. La journée du 6 est réservée aux ateliers et sessions avancées. Site : <http://www.cftl.fr/JFTL/accueil/>

23 & 24 : Cloud Expo & IoT World

Les salons dédiés au Cloud computing et aux IoT

LES MEETUPS PROGRAMMEZ!

Les meetups annulés seront reprogrammés à de nouvelles dates dès que possible.

16 juin : Sujet à venir

A PARTIR DE 18H30

Où : Infeeny 5 rue d'Uzès Paris
Métro : station Grands Boulevards (lignes 8 & 9)

Informations & inscription : programmez.com

Octobre

30 : Agile Tour Brussels/Bruxelles

C'est l'événement agilité en Belgique. +200 personnes attendues. Site : <http://www.agiletourbrussels.be>

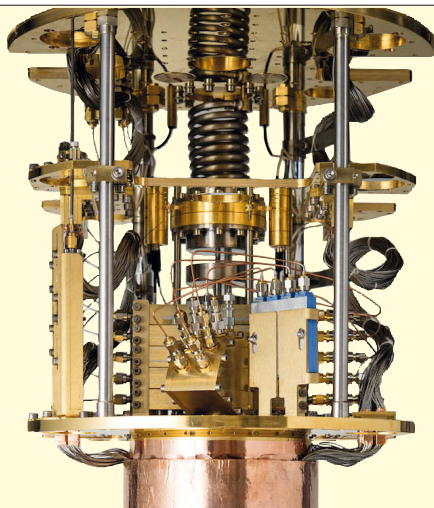
Evénements en ligne

Plusieurs salons ont annoncé une transformation en événement en ligne. Les dates restent à préciser.

- GitHub Satellite 2020
- Microsoft Build : mai
- WWDC 2020 : juin

Reporté à 2021

- MixIT/Lyon : De nombreux thèmes sont abordés : design, technologie, makers, éthique dans l'IT, style de vie, travail en équipe, etc. <https://mixitconf.org>
- RGP Level 3 : 20&21 mars 2021
- BreizhCamp 2020
- Google I/O
- Facebook F8
- FlutterConf Paris
- Best of web
- RivieraDev
- DevOxx



ÉVÉNEMENT QUANTIQUE

24 septembre 2020

où : Epitech Paris

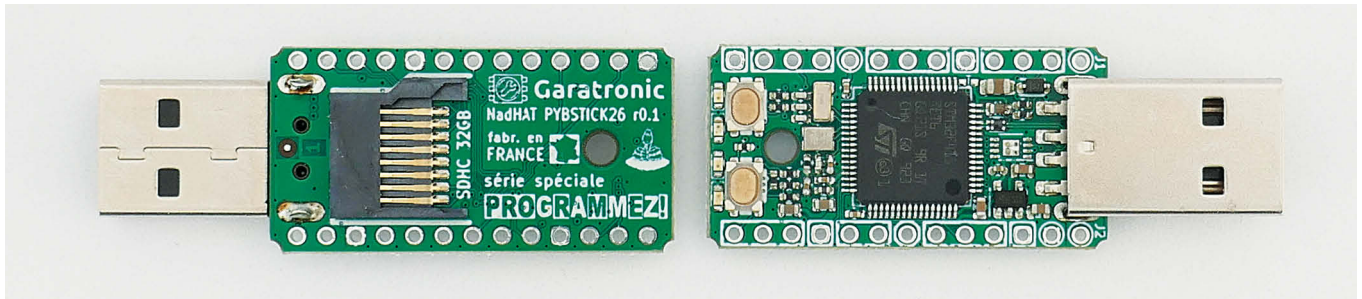
Informations : www.programmez.com

Merci à Aurélie Vache pour la liste 2020, consultable sur son GitHub :

<https://github.com/scraly/developers-conferences-agenda/blob/master/README.md>

Disponibilité :
fin juin*

Carte IoT / Maker **Programmez!**

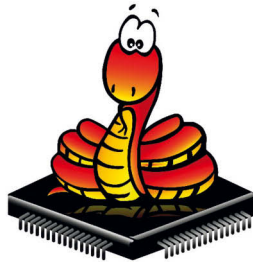


Programmez!, Garatronic et MC Hobby

s'associent pour vous proposer
une plateforme de développement **puissante**,
compacte et **pas chère**.

Elle utilise la PYBStick et le langage MicroPython.
Compatible Arduino**.

PROGRAMMEZ!
Le magazine des développeurs



* disponibilité prévisionnelle. La disponibilité réelle peut subir un retard
** les fonctionnalités de la carte seront annoncées à la sortie



shop.mchobby.be
garatronic.fr

Garatronic♥MicroPython♥MC Hobby
Control your world in a Pythonic way!

Roadmap des langages

Chaque mois, *Programmez!* vous propose un panorama des sorties des versions des langages, frameworks, etc.

Le Coronavirus a/aura des conséquences sur les roadmaps de certains langages mais il est encore difficile de les estimer.

Symfony 5.1

Mai

Une des grosses nouveautés est l'apparition du mode safe http Preference via Prefer: Safe http. Cette fonction sera implémentée dans le composant HttpFoundation. On verra aussi l'apparition des propriétés typées dans PropertyInfo. Elles sont apparues dans PHP 7.4.

2^e semestre 2020/2021

Python 3.9.0

Été/automne

Les premières versions alpha de la 3.9 ont été distribuées fin janvier. La phase bêta est attendue vers mai/juin. Désormais, il faudra s'attendre à une mise à jour annuelle du langage. Bref : moins de nouveautés, mais plus d'évolutions et d'améliorations. Parmi les nouveautés attendues : des améliorations sur le garbage collector et la résurrection d'objets (pouvant provoquer des blocages), évolution de la stabilité ABI pour éviter les incompatibilités sur les différents systèmes.

Java 15

Automne

Java 15 supportera les blocs de texte. Cette fonctionnalité est attendue depuis les premières préversions sorties avec Java 13. Les blocs de texte sont des littéraux chaînes à plusieurs lignes. C'est tout simple, mais cela permet d'insérer beaucoup plus facilement, dans le code Java, des extraits de code en HTML, XML, SQL, JSON, etc.

Java 15 fera aussi du ménage en retirant Nashorn, le moteur JS d'Oracle. GraalVM rend obsolète le moteur.

Swift 5.3

?

Une des grosses nouveautés de la v5.3 sera le support officiel de Windows. Eh oui ! Le langage sera enfin disponible sur ce système. La partie Linux sera étendue à de nouvelles distributions. Cependant, se posera la question du support Windows de certains SDK Apple comme SwiftUI. Au-delà, la version 6 apportera pas mal de nouveautés et d'améliorations :

- Possibilité de porter le langage sur de nouvelles plateformes (sur ARM ?) ;
- Des améliorations sur le gestionnaire de paquets ;

- Temps de compilation réduits ;
 - Meilleure autocomplétion du code.
- Aucun agenda n'a pour le moment été communiqué.

PHP 8.0

Fin 2020 ?

La prochaine grande version de PHP doit arriver fin 2020 si tout va bien. Tout n'est pas encore fixé, mais nous connaissons les premières nouveautés et évolutions du langage :

- Forte possibilité d'une casse de compatibilité avec les versions 7.x : sur ce point, on attend les précisions ;
- Les types Union vont arriver avec PHP 8, qui permettront de définir plusieurs types pour les arguments reçus par une fonction, ainsi que pour la valeur qu'elle retourne. En plus du type self, le type static va devenir un type de valeur retournée valide ;
- Le compilateur JIT de PHP 8 promet d'importantes améliorations de performances.

TypeScript 3.9

?

L'équipe TypeScript promet des améliorations sur les performances, le temps de compilation. La v3.9 doit corriger un comportement anormal sur certaines déclarations de fonctions. Le processus d'inférence de la 3.9 doit corriger cela.

Kotlin 1.4

?

La principale nouveauté est un nouvel algo d'inférence de type (en préversion dans la 1.3). Le nouvel algorithme d'inférence déduit des types dans de nombreux cas où l'ancienne inférence vous obligeait à les spécifier explicitement. Par exemple, dans l'exemple suivant, le type du paramètre lambda est-il correctement déduit comme étant String?

```
val rulesMap: Map<String, (String?) -> Boolean> = mapOf(
    "weak" to { it != null },
    "medium" to { !it.isNullOrEmpty() },
    "strong" to { it != null && "[a-zA-Z0-9]+$".toRegex().matches(it) }
)
```

La nouvel allocateur d'objets fonctionnera jusqu'à deux fois plus rapidement sur certains benchmarks, selon JetBrains. Cette version inclura aussi un

nouveau compilateur IR (intermediate representation) côté backend Kotlin / JS. Mais l'éditeur prévient qu'il y aura un problème de compatibilité binaire

Flutter

Pour cette année, les équipes Flutter vont travailler sur plusieurs éléments : les éditeurs web et desktop. L'objectif est de disposer d'une version web au niveau des versions mobiles. Dès cette année, le but est pouvoir créer des applications tournant sur les navigateurs, macOS, Android, Windows, iOS et... Fuschia (l'OS qui doit fusionner tous les modèles d'OS de Google). L'autre défi est d'améliorer la qualité de la plateforme et du code, réduire les bugs et de fixer les bugs remontés.

Angular 9.1

Disponible

Angular 9.1 ajoute le support de TypeScript 3.8 et propose des builds plus rapides. TypeScript 3.8 apporte pas mal d'évolution dans le langage, notamment en lien avec ES2020.

React Native 0.62

Disponible

Cette version supporte Flipper. Il s'agit d'un outil de debug des apps mobiles. Il propose un inspecteur réseau / base de données, du reporting sur les crashes des applications. La v0.62 étend le mode sombre, modification du support Apple TV, le mécanisme de mise à jour de React Native subit aussi des améliorations. Attention : plusieurs casses de compatibilités existent (retrait de propTypes, accessibilityStates, modification sur le TextInput).

Bootstrap 5.0

?

Le framework bien connu ne supportera bientôt plus IE 11 ! Ce retrait n'est pas une surprise en soi. La v5 devrait aussi retirer le support de jQuery au profit de Vanilla JS. Ce changement ne se sera pas sans conséquence sur les projets en production.

Parmi les autres nouveautés prévues :

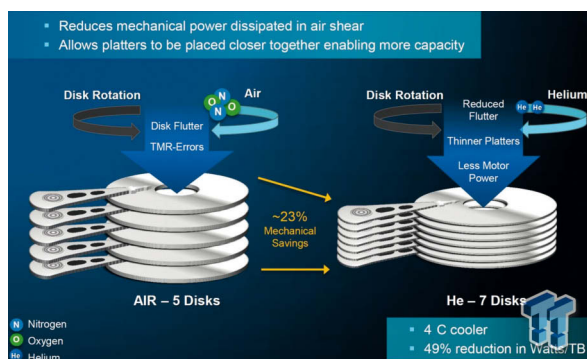
- Documentation écrite avec Hugo
- Retrait de QUnit
- Changement sur CSS
- Infra de tests s'appuyant sur Jasmine



François Tonic

Disque dur, SSD : bien choisir !

On nous parle de stockage en ligne, de partages de fichiers comme avec Dropbox mais le stockage physique local n'a pas disparu, loin de là. Les deux approches sont conseillées pour avoir une stratégie de réplication des données proactive. Un crash, une panne réseau, une corruption de données ne sont jamais à exclure.



Et ne pensez pas que les erreurs disques n'arrivent qu'aux autres. J'ai dernièrement eu une corruption partielle que nous n'avions pas immédiatement identifiée. Or, le disque en cause provoquait une instabilité sur le système entier. J'ai découvert le problème suite à une erreur backup. Et en faisant une analyse du disque, les erreurs se sont multipliées. Or, il s'agissait un WD Red de 2 To à destination des NAS. Aussi fiable soit-il, une corruption peut survenir, pas forcément sur les disques physiques mais sur le système de fichiers utilisé. Et encore, j'utilise un file system journalisé mature et stable. Heureusement, pour moi, ce n'était qu'un de mes disques de backups, déployés depuis +3 ans, le système alterne de disque à chaque sauvegarde (en utilisant le delta depuis la dernière sauvegarde sur ledit disque), soit x fois par jour.

Pour remettre à zéro le disque, j'ai décidé d'utiliser le puissant utilitaire Windows :

diskpart. Opération très longue mais souvent efficace en formatage de bas niveau. J'ai pu récupérer plusieurs disques devenus totalement illisibles depuis l'OS ou le gestionnaire NAS. Cependant, ce n'est pas une solution miracle pour tous les problèmes.

Des disques durs très différents

Il n'existe pas un type de disque dur mais plusieurs, selon les usages. Et les prix peuvent rapidement doubler ou tripler. Seules certaines gammes permettent des capacités au-delà de 10 To.

Si vous prenez Western Digital, il existe des gammes pour le NAS, le desktop, la vidéo-surveillance, le serveur / datacenter. Pour les NAS et les sauvegardes régulières, sans NAS, la gamme Red est un bon choix. Le Red Pro est plus performant. En revanche, les Red ne sont pas adaptés à un usage quotidien dans un desktop. On retrouve la même segmentation chez Seagate avec les Barracuda (desktop), Firecuda (jeux) et IronWolf (NAS).

Si vous prenez des disques à haute densité, au-delà de 8 To, optez pour les gammes serveurs / NAS. Ces gammes sont souvent équipées de systèmes anti-vibrations et certains modèles utilisent même de l'hélium pour réduire les résistances dans l'air et les frottements. Mais attention on reste sur du mécanique, donc avec des pannes possibles.

SSD : le plus performant

Le disque SSD est partout et s'est rapidement imposé, avec la baisse des prix sur les densités élevées (+1 To). Là aussi, la qualité peut être variable. Certains modèles sont plébiscités pour leurs performances et leurs stabilités dans la durée. Que ce soit en interne ou externe, tous les SSD ne se valent pas. Et les performances sont cruelles ! Actuellement, les NVMe sont les plus per-

ATTENTION : UN DISQUE TOMBE AUSSI EN PANNE

Soyons clairs : même un disque très haut de gamme peut tomber en panne 2 jours après son installation. Ce n'est pas fréquent mais cela arrive. Dès les premiers signes de faiblesses, de bruits anormaux, demandez l'échange standard. Ce problème peut aussi arriver avec un SSD, même si ces pannes sont très rares.

Le 20 mars dernier, HPE communiquait sur un très vilain bug : des SSD peuvent littéralement devenir inaccessible, sans aucun accès aux données après 40 000 heures de fonctionnement ! Une mise à jour urgente du firmware est nécessaire pour fixer le bug... En novembre 2019, un autre bug du même genre avait déjà été publié pour certains SSD après 32 768 heures de fonctionnement. En 2012, Crucial avait eu un joli bug sur le modèle M4 : après 5184 heures, le SSD plantait toutes les heures... Support HPE :

https://support.hpe.com/hpsc/public/docDisplay?docLocale=en_US&docId=a00097382en_us

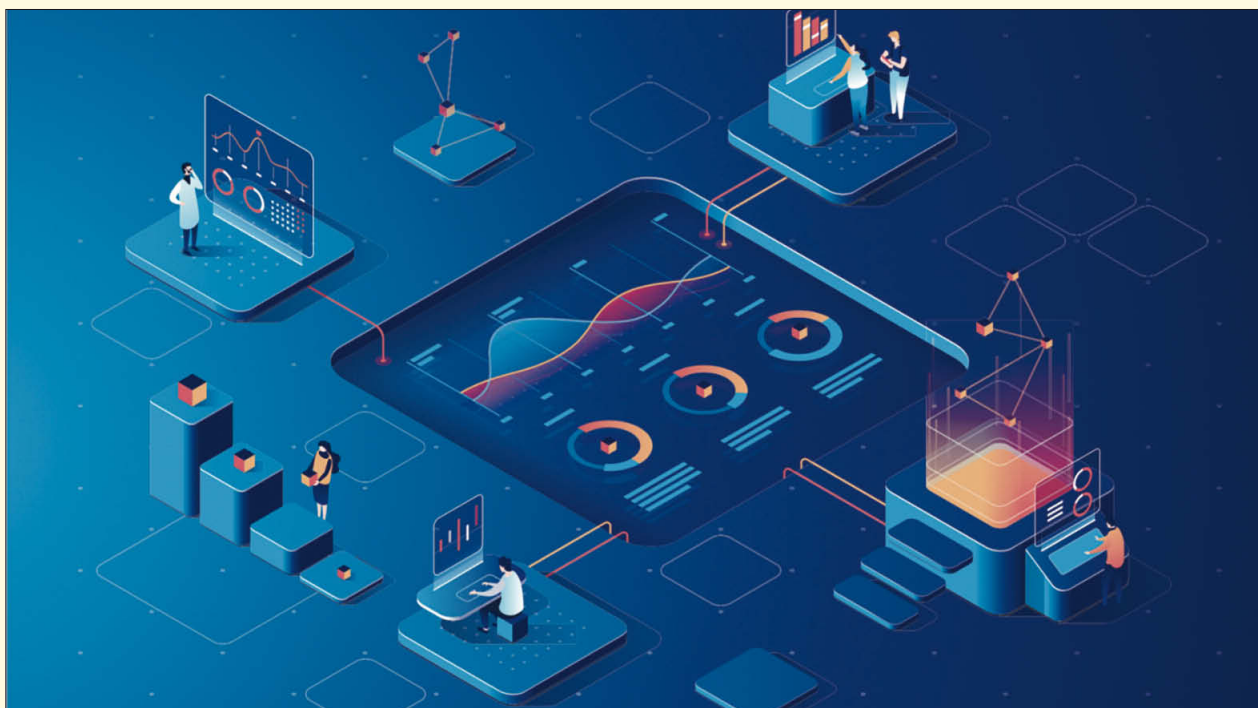
Soyez très vigilant sur vos disques en NAS, surtout quand ils sont en RAID. En RAID 0, vous n'aurez pas de tolérance de panne, si un disque tombe, tout tombe.

formants mais attention à bien choisir la connectique. Un full Thunderbolt 3 sera très performant. Un USB-C suffira dans la plupart des cas mais vous perdrez en performances lecture / écriture. Un conseil : n'oubliez pas le dissipateur ou la pâte thermique si vous optez pour une barrette NVMe + boîtier. L'ensemble dégage rapidement de la chaleur.

Les Samsung série 970 sont parmi les barrettes NVMe les plus performantes du marché, même si les tarifs sont élevés.

Dix ans d'Open Source pour Alibaba

En 10 ans, le groupe Alibaba et sa branche data intelligence, Alibaba Cloud sont devenus des acteurs majeurs de la communauté Open Source, contribuant à plus de 1 700 projets et faisant don de certaines de leurs innovations à des fondations comme Apache.



Lorsque Alibaba s'est lancé dans l'Open Source, il y a tout juste 10 ans, l'ambition initiale était modeste. Alibaba voulait partager des codes qui pourraient être utiles à d'autres et pensait qu'il serait intéressant de construire en même temps une communauté de contributeurs partageant les mêmes idées. À partir de ce postulat, Alibaba et Alibaba Cloud sont arrivés à mettre en place un modèle de collaboration à grande échelle et à haute performance, à même d'optimiser la technologie. Indépendamment de la langue ou de la race, l'esprit

d'ouverture, de partage, d'égalité et d'intégration que représente l'open source a efficacement comblé le fossé technique et a été un véritable moteur de développement.

Dix ans plus tard, Alibaba a contribué à plus de 1700 projets open source, couvrant des domaines allant du big data aux logiciels cloud natifs, en passant par l'IA, les bases de données, les middlewares et le matériel. Alibaba Cloud a reçu plus de 750 000 étoiles sur GitHub, et plus de 20 000 contributeurs ont travaillé sur ses projets open source. « Nous n'aurions pas

pu réaliser ce que nous avons aujourd'hui sans la contribution et la confiance des développeurs du monde entier. En 2017, OpenMessaging, dont nous sommes l'un des premiers instigateurs et contributeurs, est par exemple devenu le premier standard international pour le calcul distribué, » déclare Dr Yangqing Jia, President and Senior Fellow de la division Data Platform division d'Alibaba Cloud Intelligence.

Au cours des dix dernières années, Alibaba est devenue l'une des principales entreprises en termes de niveau de collaboration avec

les communautés open source. Elle est membre de plus de dix fondations Open Source, tant en Chine qu'à l'étranger, dont Java Community Process (JCP), Linux, RISC-V, Hyperledger, MariaDB et OCI. Le groupe a également fait don de quatre projets de premier ordre à Apache, et plus de 10 des projets d'Alibaba ont été inclus dans le paysage de la Cloud Native Computing Foundation (CNCF).

Le Dr Yangqing Jia partage avec nous la stratégie de l'Open Source qui a fait le succès d'Alibaba dans le domaine.

LA COMMUNAUTÉ OPEN SOURCE EST UN BERCEAU POUR LA COLLABORATION ET L'INNOVATION

*Interview du Dr Yangqing Jia, Vice-Président d'Alibaba Group,
Président et Senior Fellow de la division Data Platform d'Alibaba Cloud Intelligence.*

Pourquoi le groupe Alibaba a-t-il rejoint la communauté open source et quand a-t-il commencé à y contribuer ? Pouvez-vous nous en dire plus sur l'étendue de vos contributions à l'open source ?

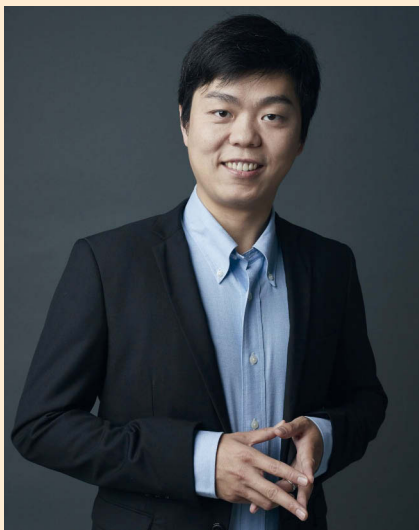
Dr Yangqing Jia : Alibaba pense que les communautés open source sont le berceau de la collaboration et de l'innovation. La société a lancé son premier projet open source en 2010. Dix ans plus tard, Alibaba a plus que contribué à plus de 1 700 projets open source. Plus de 20 000 développeurs ont contribué aux projets d'Alibaba, leur attribuant plus de 750 000 GitHub Stars. Alibaba est d'ailleurs l'un des dix premiers contributeurs à GitHub.

Pouvez-vous citer quelques projets open source clés d'Alibaba ?

Dr Yangqing Jia : Alibaba a fait don de quatre projets de premier ordre, Dubbo, RocketMQ, Weex et JStorm qui sont devenus des projets majeurs de la fondation Apache. D'autres projets clés sont par ailleurs en cours, comme Alink, notre plateforme d'algorithme de machine learning basée sur Flink, ou OpenMessaging la première norme internationale pour le calcul distribué. Parmi les autres projets Open Source importants, nous avons également DragonFly, un système de distribution d'images natives dans le cloud qui a été inclus dans un projet de la CNCF au niveau sandbox en 2018.

Préférez-vous contribuer aux projets existants ou lancer vos propres solutions en open source ?

Dr Yangqing Jia : Nous faisons les deux de manière exponentielle. Par exemple, en octobre 2019, Alibaba Cloud et Microsoft ont annoncé l'Open Application Model (OAM), un standard ouvert pour le développement et l'exploitation d'applications sur Kubernetes et d'autres plateformes. En novembre 2019, nous avons également ouvert Alink sur GitHub. Nous mettons également à disposition des développeurs AliOS Things, la plateforme intégrée utilisée par tous les appareils



Jia Yangqing, Vice-Président d'Alibaba Group, Président et Senior Fellow de la division Data Platform d'Alibaba Cloud Intelligence rappelle qu'en 10 ans, Alibaba a participé à plus de 1700 projets Open Source.

professionnels Alibaba Cloud IoT, et Atlas. Elle offre un cadre de développement Android flexible permettant de résoudre les problèmes de coopération d'équipes à grande échelle. Elle convient au développement d'applications de petite ou grande taille en responsive design sur toutes les versions d'Android supérieure à la 4.0.

Que pensez-vous de l'intérêt des développeurs pour l'open source ? Est-ce qu'ils varient en fonction du fait qu'ils travaillent individuellement ou pour de grandes organisations comme Alibaba ?

Dr Yangqing Jia : Tous ont un intérêt plus ou moins élevé, les développeurs travaillant individuellement, avec des PME ou pour de grandes entreprises technologiques peuvent tous bénéficier de l'open source. Par exemple, Alink est une plateforme d'algorithme de Machine Learning basée sur Flink et utilisée notamment dans le cadre du festival mondial du shopping du 11.11. En novembre 2019, lorsque nous avons lancé Alink en open source sur GitHub, nous avons reçu de nombreuses contributions de la communauté, de tous types de développeurs, et les avons utilisées. Ainsi, les

développeurs des grandes organisations peuvent également bénéficier de la communauté open source.

Alibaba a-t-il davantage bénéficié de l'Open Source ? Où y a-t-il plus contribué ?

Dr Yangqing Jia : J'estime que nous bénéficions beaucoup de la communauté open source, et inversement. C'est pourquoi nous nous engageons à travailler avec elle le plus tôt possible dans nos cycles de développement de logiciels. Nous travaillons en étroite collaboration avec des fondations open source. Alibaba est un membre clé de plus de 10 fondations open source, tant au niveau national qu'international, dont la Linux Foundation, la Cloud Native Computing Foundation (CNCF), l'Apache Software Foundation, la MariaDB Foundation, la RISC-V Foundation, le Confidential Computing Consortium, la Reactive Foundation, l'Open Container Initiative, la CD Foundation et l'Alliance for Open Media.

En octobre 2019, Alibaba Cloud a annoncé un partenariat avec MongoDB pour fournir un service managé basé sur leur solution de base de données. Pourquoi ne pas simplement utiliser la version open source et le service de compilation ? Pouvez-vous faire le point sur le fonctionnement de ce service MongoDB ?

Dr Yangqing Jia : Nous avons annoncé notre partenariat avec MongoDB en octobre 2019 pour offrir aux entreprises clientes une solution MongoDB-as-a-service autorisée. MongoDB est une base de données populaire dans le monde entier, et avec la dernière édition Community et Enterprise Advanced de MongoDB fonctionnant sur notre plateforme, nous sommes en mesure de répondre à un large éventail de besoins des clients, des développeurs open source aux équipes informatiques d'entreprises de toutes tailles. Nous nous sommes également engagés à aider nos partenaires à construire des produits solides sur notre plateforme PaaS afin que nous puissions innover ensemble et contribuer en retour à l'écosystème au sens large.



François Tonic

Télétravail, Coronavirus : urgence et méthodologie

La crise du Coronavirus a bouleversé la vie quotidienne de plusieurs milliards de personnes dans le monde entier. Mais pas de la même manière partout. Confinement, sorties limitées, de nombreuses personnes ont dû improviser en quelques jours, voire quelques heures, le télétravail. Le métier de développeur est un des métiers qui s'y prête le mieux, normalement.

Le coronavirus a forcé de nombreux salariés à travailler depuis leur domicile. Cette circonstance exceptionnelle est prévue par le code du travail : le télétravail peut être imposé à un salarié même sans accord. Et si le poste occupé le permet, le télétravail est à utiliser. Enfin le télétravail peut être mis en place pour assurer la continuité de l'activité de l'entreprise et garantir la protection des salariés. Tous les postes ne sont pas éligibles.

Trois éléments sont importants si vous travaillez en équipe :

1. organiser son travail ;
2. travailler en équipe via les outils de collaboration et de partages ;
3. communiquer.

La définition des tâches, leur gestion et leur réalisation sont essentielles dans les projets. En télétravail, cela ne change pas. Le remote ne doit pas compromettre les développements, ni tous les pipelines d'intégration, et de tests (typiquement tout ce qui CI/CD).

L'exemple de Voodoo

Voodoo est un éditeur de jeux mobiles. Nous avons échangé avec Vincent Vallet, développeur back-end chez Voodoo. Pour l'éditeur, le télétravail n'est pas une nouveauté. Il est proposé 1 journée par semaine, non obligatoire. Au niveau organisation, il ne faut pas que l'absence puisse empêcher le développement des équipes et des projets. « C'est une journée comme une autre dans son organisation » évoque Vincent. « Certaines tâches nécessi-

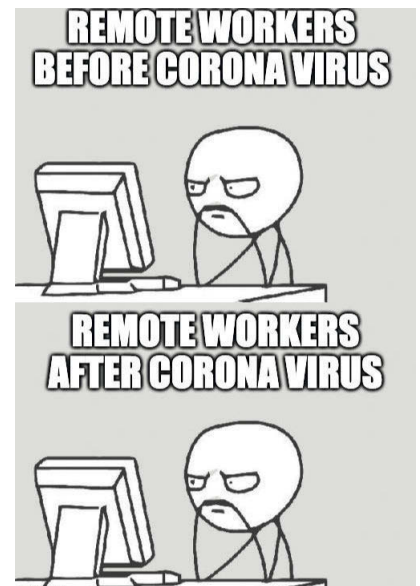
tent du calme. Parfois, nous avons le bruit de l'open space, les gens qui viennent tout le temps vous voir. Pour moi, quand je fais de l'algo, cela peut demander plusieurs heures à se concentrer et dans le calme ».

Une des difficultés est de créer des « rites », une culture d'entreprise prenant en compte le télétravail. Ainsi, les réunions quotidiennes restent la norme, même à distance. Tous les outils existent pour le faire. Côté horaire, la réunion quotidienne se déroule généralement à 10h30. Mais après, la journée peut s'organiser un peu comme on veut et selon les tâches à coder.

Côté outils et organisation des projets, « nous avons tous les outils pour être indépendants comme GitHub, les infrastructures de type AWS, les services web » évoque Vincent. La connexion aux serveurs et outils de l'entreprise se fait en SSH. Pour gérer les tâches, Voodoo utilise Trello et Slack pour la communication entre les équipes. Mais les équipes sont assez indépendantes et donc elles peuvent choisir leurs propres outils. Mais une unification de tous ces moyens de communications et d'échanges est envisagée. Plus vous avez d'outils, plus vous devez réaliser des interconnexions, ce qui n'est pas toujours le plus simple.

Côté matériel, « nous sommes tous sur portables. Quand on est en télétravail, on le prend avec nous. Parfois, on peut travailler sur son matériel perso » explique Vincent.

Quel retour du télétravail pour Vincent ? « Sur le quantitatif, c'est difficile à dire. Par-



fois, il faut être avec le reste de l'équipe. Plus simple et plus sympa. Pour des tâches individuelles, parfois 1 journée c'est un peu juste. Cela dépend vraiment des développements à faire. Venir au bureau de 9h à 18h ce n'est pas forcément là que l'on est le plus productif. » explique Vincent.

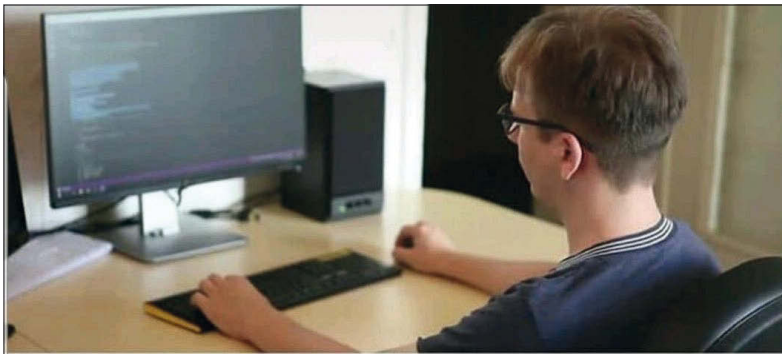
Parfois, quand on bloque sur un problème, on prend l'air 1h et je reprends le point bloquant. Cela peut aider à trouver une solution ou voir le problème autrement.

Pour Vincent, le confinement est plus une extension. Et cela peut aussi aider à connaître les idées reçues sur le télétravail. Cependant, attention, il faut aussi une discipline, une rigueur. « C'est un point délicat. Quand on commence le télétravail, la rigueur n'est pas facile. Ce n'est pas un environnement de travail. Il faut de la rigueur, de l'organisation, et maintenir les rites » précise Vincent.

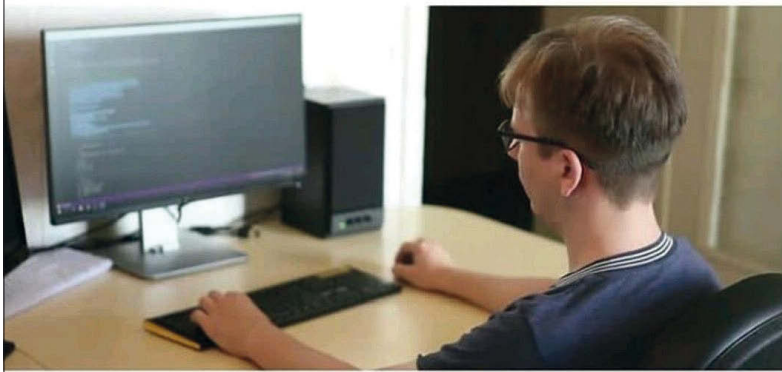
L'exemple de Yoozio

Le coronavirus impacte directement les entreprises, les salariés et donc les développeurs qu'ils soient en entreprise, en ESN ou indépendants. Nous avons interrogé Sébastien Houzet de Yoozio, petite ESN travaillant beaucoup dans le e-commerce et la transformation numérique, essentiellement pour les grands comptes.

Pour Sébastien, le télétravail n'est pas une



A programmer



A programmer in quarantine

nouveauté au quotidien. Il est même une règle depuis plusieurs années. La grève dans les transports a eu un impact limité en décembre et janvier. La crise actuelle a mis l'ensemble des développeurs en télétravail. « Dans le développement, nous avons tous les outils en ligne nécessaire pour coder. Nous n'avons pas de lien avec des serveurs physiques dans nos bureaux. Nous utilisons des outils en ligne pour gérer le code, la documentation, etc. Nous nous assurons que les développeurs puissent avoir accès de n'importe où aux réseaux des clients via un VPN, des IP spécifiques, etc. C'est un élément important. » précise Sébastien. Côté outils, Yoozio n'impose pas même si de facto, les développeurs possèdent des outils de base tels que Visual Studio Code, Sequel Pro (pour la partie MySQL), Git. La partie communication en télétravail est un élément crucial. Il faut pouvoir échanger, communiquer, expliquer. Selon l'urgence de la demande, les canaux de communication ne seront pas les mêmes. « Si nous avons besoin d'une réponse rapide et directe, nous avons plutôt utilisé Slack. Si nous sommes dans l'urgence, ce sera le téléphone. Pour les réunions, nous

utilisons beaucoup Google Meet. » poursuit Sébastien. Le mail est très peu utilisé par les équipes et son usage est limité et les clients sont incités à les réduire dans la mesure du possible.

Yoozio n'utilise pas un seul outil, mais plusieurs. Une réflexion se fait jour pour déployer un environnement unique pour intégrer efficacement tous ces canaux. Discord a été testé. La notion de room est très pratique. La question d'équiper les dévs de casques audio et de micros de qualité est en réflexion, car parfois, durant les visioconférences et les appels, les écarts de qualité peuvent se ressentir. Pour Sébastien, il manque encore un bon tableau blanc virtuel pour partager les croquis, les architectures.

Sur les horaires, l'ESN est très souple. Un élément important est que l'absence d'un développeur ne doit impacter un autre dév. Il faut faire confiance aux développeurs.

« On essaie de créer des rituels, par exemple, une réunion virtuelle à 10h chaque jour, pour discuter des projets ou de tout autre sujet, créer une ambiance. » Indique Sébastien.

Tous les développeurs utilisent des

1st rule of programming:-

If it works don't touch it.

Rule of thumb for programmers

NE PAS OUBLIER LE MATÉRIEL !

Le poste de développement est l'outil quotidien du développeur. Dans le cas du confinement, déplacer en urgence un desktop avec tout ce qui avec, ce n'est pas toujours simple.

Un bon portable (configuration optimale pour le développement bien entendu) remplacera le desktop. Un écran, ou plusieurs externes permettent une réelle souplesse de travail. En cas de besoin, un kit de sécurité peut être rajouté.

Les autres points à surveiller :

- La connectivité aux réseaux
- Quelle sécurité pour les accès serveurs / données internes ?

Nerf non inclus dans le pack télétravail...

machines fournies par l'ESN. Sur la partie réseau et débits, Sébastien précise que la plupart des dévs sont dans des zones ayant de très bonnes connexions. Au pire, l'usage de la 4G pourra pallier les débits plus faibles. Parfois, pour économiser de la bande passante, la webcam est désactivée durant les réunions virtuelles.

La productivité est une question récurrente en télétravail. Il faut faire confiance aux développeurs. Mais gérer la famille, les enfants, les animaux, c'est loin d'être facile. Avec les enfants, selon l'âge, il faut notamment les occuper et s'assurer qu'ils fassent les cours et devoirs envoyés par les écoles, depuis la fermeture de toutes les écoles. Forcément, la productivité baisse.

Télétravail = productivité = oui, mais non !

Si on consulte des études, nous serions dans un monde merveilleux avec +80 / 85 % de productivité et une qualité de travail. Dans la réalité, ce n'est pas du tout ça. De



nombreux facteurs jouent. Le télétravail exige, comme l'a dit Vincent, de la rigueur, une organisation claire. Car on peut vite partir en vrille et finalement être dépassé par le travail à faire ou tout simplement, faire l'inverse du télétravail.

Certaines entreprises imposent des horaires, du moins, une présence derrière le poste. N'oubliez pas que vous êtes tout de même en entreprise même si c'est chez vous. Certains métiers sont plus adaptés que d'autres. Et tout le monde n'est pas « remote compatible ». Et nous l'avons constaté durant tout le confinement.

La dérive est aussi une confusion croissante entre pro et perso : on commence à bosser le soir, seulement à partir de 12h ou 14h, un peu le week-end, car je n'ai pas fini en semaine, etc. C'est un risque qu'il ne faut pas sous-estimer.

Communiquer oui mais attention à l'abus de communication !

On dispose de nombreux outils de communication, de partages documentaires, de gestion de présences, de planifications, etc. Réduire au minimum les outils nécessaires, dans l'idéal, essayez de déployer une solution unifiée pour éviter de multiplier les outils.

Comme vu plus haut, il ne faut pas abuser des outils de communication. Selon la criticité ou l'urgence de la demande, ciblez tel ou tel outil de communication. Évitez au

maximum le mail et encore plus le téléphone. Ils consomment beaucoup de temps.

Le planning, les agendas, la définition et le suivi des tâches : soyez très rigoureux dessus. Un outil comme Trello est là pour ça. Mais attention, là encore, allez à l'essentiel, ne surchargez pas inutilement.

Pour les chefs de projets, CTO, Tech Lead, etc., n'oubliez pas de créer une ambiance, de communiquer avec vos équipes. Si des collaborateurs sont absents pour maladie, soyez transparents. Une des difficultés est de pouvoir créer une dynamique de groupe, de garder le contact et de continuer le travail. L'accès aux outils, aux sources, aux documentations sont essentiels.

Si vous avez des enfants à maison, le télétravail n'est jamais simple, encore moins, avec un confinement et la fermeture des écoles. Mais il faut aussi qu'ils puissent comprendre la situation et que vous avez des moments pour eux et des moments de travail.

Dans des projets lourds, prioriser les tâches critiques ou importantes. Des développements « secondaires » peuvent attendre s'ils ne pénalisent pas les itérations et les builds. N'oubliez pas de savoir si les utilisateurs, le client du projet, sont disponibles et continuent à fonctionner. C'est utile si vous avez des questions sur une fonctionnalité ou une build à envoyer pour tests ou tout simplement pour de la pré-prod ou prod.

PETITE FAQ

Cette FAQ est fournie à titre indicatif et n'a aucune valeur légale. À vous d'interroger votre responsable RH / employeur.

Contrat de travail

En situation normale, le télétravail doit être indiqué dans votre contrat de travail. Vous êtes toujours rattaché(e) à l'entreprise avec tout ce qui va avec. Cela signifie que les assurances de l'employeur couvrent l'activité. Même si tout le monde est OK pour le télétravail, disons 1 ou 2 journées par semaine, mieux vaut modifier le contrat de travail. Cela évite, pour chaque parti, des complications en cas de conflits ou une mésentente sur l'interprétation du télétravail.

Et les assurances ?

Comme dit plus haut, les assurances de la société couvrent en télétravail. Une assurance multirisque informatique (pour le matériel) doit être prise par la société. Si besoin, vous devrez fournir une attestation de votre assurance multirisque habitation.

Faut-il une autorisation du propriétaire ?

Si vous êtes locataire, vous devez, théoriquement, demander l'autorisation au propriétaire pour exercer une activité professionnelle, même ponctuelle, au domicile. En situation normale.

Prise en charge par l'employeur

La société doit prendre en charge tous les coûts liés au télétravail (matériel, logiciels, abonnements spécifiques, téléphone, etc.). Par exemple : l'utilisation de la connexion internet personnelle doit être remboursée au prorata de l'utilisation professionnelle faite (sortez les équations pour le calcul). Les m2 occupés par le salarié en télétravail ne sont pas, en principe, indemnisés par la société.

Limite de l'utilisation des matériels de l'entreprise ?

Le matériel informatique provenant de l'entreprise n'est pas laissé au libre usage du salarié. À charge à l'employeur d'indiquer les usages et les limites d'utilisation et les sanctions éventuelles.



Frédéric BOUVARD
CTO - Finance Active
<https://stackshare.io/finance-active/finance-active>

Les premiers jours de la crise Coronavirus vus de l'intérieur chez Finance Active.

Lundi 02 mars 2020 : comme beaucoup d'entreprises, on se dit qu'il faut commencer à prendre la menace sanitaire au sérieux, on commence à donner et relayer les 1ères consignes d'hygiène, de limiter les déplacements, d'effectuer 2 semaines de quarantaine si on a côtoyé une personne avérée ou si on est allé dans une zone à risque..., etc. Les RH sont sur le pont et ne le quitteront plus. Le 5 mars, tous les déplacements des équipes de R&D sont annulés par défaut et je les validerai au cas par cas. On coupe notamment le pont (ferroviaire) entre nos équipes Paris et Nantes.

Semaine du 09 mars 2020 : je commence à me dire qu'il faut préparer le passage au stade 3... Mais c'est le quoi le stade 3 ? Pas clair... Transports ou pas transports ? On ne sait pas... Rien dans les textes, mais il faut anticiper pour ne pas être pris au dépourvu. A la R&D de Finance Active, le télétravail on connaît, chaque employé a le droit à 2j de télétravail par semaine. On a tous l'outillage collaboratif qui va bien (Teams et Jira). On est organisés en petites équipes Agile. Les rituels sont bien en place. Pourtant je ne souhaite pas passer tout de suite en full remote. On sort déjà d'un mois de grèves dures avec beaucoup de remote. Il y a quand même besoin de remettre un peu de liant en présentiel... Je pars donc sur un mode où chaque équipe sera coupée en deux pendant la durée d'un sprint. 1 sprint en remote, 1 sprint en présentiel, en gardant toujours les 2j de télétravail... En gros, cela fait 6j sur site par mois par personne. Cela permet ainsi d'espacer les collaborateurs dans l'open space.

Jeudi 12 mars, le plan est présenté à la R&D, quelques questions mais rien de bloquant le plan est compris et adopté... Et là, tout s'accélère...

Jeudi soir, Emmanuel Macron s'adresse aux français. Le discours est grave, mais pour autant, la France n'est pas déclarée en stade 3.

Vendredi 13, je décide d'anticiper et d'appliquer le plan stade 3 sans attendre, à démarrer lundi 16 mars. En parallèle avec l'IT on s'assure que le VPN tiendra le choc. Car au-delà des 50 membres de la R&D et du Produit, chez Finance Active, il y a environ 100 autres personnes (Consulting, finance, RH, com...) qui ont besoin d'accéder à différentes ressources internes à l'entreprise... La crise arrive un peu trop tôt. On est en pleine migration de l'informatique interne vers AWS d'une part et vers le Sharepoint pour tous les fichiers partagés des équipes de Consulting d'autre part mais le timing est trop juste, on n'est pas prêt à basculer en quelques jours.

Samedi 14, Muriel Penicaud annonce que le télétravail n'est pas une option mais doit être appliqué à toutes les entreprises qui le peuvent... Là il n'y a plus de doute, je jette à la poubelle le plan stade 3.

Dimanche 15, je publie sur le canal principal Teams de notre R&D le full remote pour tous et l'interdiction absolue pour notre équipe de Prod de se déplacer (notre offre applicative est entièrement SAAS). Le but est de sanctuariser le run, même si on se doute que nos clients risquent d'avancer en ordre dispersé. On se doit d'être up & running. Notre équipe IT, quant à elle, restera lundi au bureau pour s'assurer que le matériel tient la charge.

Lundi 16 mars : 8h45, call de crise avec le Comex où l'on décide notamment d'avoir 2 points par jour (matin & soir) pour voir comment se met en place le full remote dans l'Entreprise. Il s'agit de

corriger au plus vite les problèmes rencontrés par les uns et les autres. 9h30, call de crise avec les Team Lead de la R&D pour vérifier que tout le monde est opérationnel, qu'il n'y a pas de craintes ou de pb particuliers.

Lundi après-midi : force est de constater que tout fonctionne bien à la R&D. Les équipes tournent presque comme si rien ne s'était passé.

Et après 10 jours de full remote ?

- Les sondes dans nos applications indiquent qu'elles sont toujours utilisées, qu'il n'y a pas de vraie baisse de productivité chez nos clients, ce qui démontre bien que le télétravail s'est quand même démocratisé en France (et en Europe) dans tous les secteurs qui travaillent majoritairement derrière un écran.
- Les équipes R&D continuent de produire comme à leurs habitudes et nous pouvons continuer de vider notre backlog. L'impact sur le delivery est très faible. On croise juste les doigts de ne pas avoir de crashes matériels qui nécessiteraient des pièces que l'on n'a pas en spare.
- Les petit soucis d'informatique interne sont résolus ou en passe de l'être. Pour les autres services de l'entreprise, on continue la migration vers Sharepoint et vers AWS

Les 2 points par jour du Comex sont maintenus où les discussions se tournent maintenant vers le futur une fois que le confinement sera levé... Après tout, notre Fintech est spécialisée dans la gestion de la dette, et la reprise pour les entreprises se fera par de la dette.

**Cédric PELLERIN**

Tombé dans un ZX81 en 1983, passionné de rétro-computing, de développement, de Logiciels Libres et d'administration systèmes. Pour le moment, expert Open Source chez Silicom.

Les systèmes d'exploitation "oubliés"

OS/2, le challenger surdoué mais malheureux de Windows

OS/2 WARP

J'ai choisi de commencer cette série d'articles par un des systèmes majeurs de l'ère IBM PC : OS/2. Pourquoi ce choix ? D'abord parce qu'en le réinstallant et en le reprenant en main, je me suis aperçu qu'on a finalement pas inventé grand-chose d'important en informatique depuis lors. Avec OS/2 Warp 4, donc en 2000, on avait déjà, entre autres :

- Une interface graphique totalement orientée objets ;
- Un système multi-tâches préemptif ;
- Une capacité d'émulation avec le support natif d'applications MS-DOS, Windows 3.x et même de certaines applications win32 ;
- Un réseau parlant nativement TCP/IPv4 mais aussi IPX/SPX et NetBIOS ;
- Une connexion vers les mainframes IBM avec un émulateur 3270/5250 ;
- Un début de virtualisation avec la possibilité d'installer un serveur Novell Netware 4+ au sein même d'OS/2.

Par ailleurs, aujourd'hui, beaucoup de logiciels libres sont portés sur OS/2 et on retrouve, parmi tant d'autres :

- GCC 8 ;
- Perl 5 ;
- Python 2.7 ;
- MySQL 5 ;
- PostgreSQL 9...

Nous reviendrons plus en détail sur ces évolutions dans la partie technique, mais vous pouvez d'ores et déjà sortir les disquettes et les CD-RW de l'armoire, ils vont servir !

Historique

OS/2 v1.0 date de 1987. Il est le fruit d'une collaboration plus ou moins houleuse entre IBM et Microsoft – à ce sujet et pour les anglophones je recommande chaudement de lire entièrement ce que raconte un ancien d'IBM : <https://www.quora.com/Why-did-IBMs-OS-2-project-lose-to-Microsoft-given-that-IBM-had-much-more-resources-than-Microsoft-at-that-time>. Techniquement il dépasse dès le départ tous ses concurrents, c'est un système 16 bits, multi-tâches pré-

emptif, capable de faire fonctionner des applications MS/PC-DOS natives, en même temps que des applications OS/2.

Cette toute première version ne propose pas d'interface graphique. Il va falloir attendre un an pour voir arriver le « Presentation Manager » avec la version 1.1. Pour le moment, le système de fichiers supporté est le même que MS-DOS : FAT 12 et 16. HPFS (High Performance File System) n'arrivera qu'un an après avec OS/2 1.2. Cette version fournit aussi un langage de script adulé par presque tous ceux qui ont eu à l'utiliser : REXX.

La version 1.3 d'OS/2 en 1991 n'apportera pas grand-chose de nouveau sur le plan technique, en revanche c'est là que Microsoft abandonne le projet et se concentre sur ce qui deviendra Windows NT. La vraie grosse innovation arrive avec OS/2 2.0 en 1992. Le système est enfin presque entièrement 32 bits mais il conserve la possibilité de faire fonctionner les applications 16 bits. Il est aussi compatible avec les applications Windows bien écrites (comprendre celles qui n'utilisent pas d'APIs non documentées) et, bien sûr, avec les applications MS-DOS. Cette version apporte une nouvelle interface graphique nommée Workplace Shell et son développement est totalement orienté objets, bien qu'écrit en pur C. C'est cette philosophie de l'objet qui fera un des points forts d'OS/2, de pair avec une stabilité légendaire, largement supérieure à celle de Windows. La version 2.1, sortie en 1993, est une version optimisée de la 2.0 et apporte le support des polices TrueType (polices vectorielles).

Encore un an passe et la version 3.0 nommée OS/2 Warp arrive dans les bacs. Elle inclut nettement plus de drivers que la version 2, une pile TCP/IP en standard et une suite logicielle nommée IBM Works incluant très classiquement un traitement de textes, un tableur, un petit gestionnaire de bases de données et un agenda personnel.

À cette époque, IBM dut faire face à la pé-

nurie d'applications spécifiques. Ainsi le pack Microsoft Office pour Windows était d'une lenteur et d'une instabilité redoutables sous OS/2. Il faut bien reconnaître qu'IBM ne faisait vraiment rien pour encourager les développements tiers sur son OS, contrairement à Microsoft qui a su fédérer rapidement toute une communauté de développeurs autour de Windows.

Sortie en 1996, la dernière version officiellement supportée par IBM, OS/2 Warp 4, ajoute le support de Java et la reconnaissance vocale. Warp 4 évoluera via une série de patches (Fixpacks) jusqu'à la version 4.5. Cette version est déclinée en deux distributions : OS/2 Warp 4 et OS/2 Warp Server. La version classique est destinée aux postes de travail et aux partages de ressources en poste à poste sur un LAN, tandis que la version Server se positionne directement en concurrent de Windows NT Server et ajoute des fonctions primordiales comme :

- Partage de fichiers et d'imprimantes ;
- Accès transparent aux ressources de serveurs Novell Netware ;
- Accès à distance par modem ;
- Sauvegarde ;
- Serveurs DHCP et DNS ;
- Centralisation de l'authentification pour les postes OS/2 mais aussi Windows, Mac et Unix ;
- Sécurisation des connexions via Kerberos ;
- Et pas mal d'autres choses...

En 1999 sort OS/2 Warp Server for e-Business, toujours en version 4.5. Il évoluera jusqu'en 2001, terminant en version 4.52.

En 2001, une société tierce nommée Serenity Systems sortira eComStation 1.0 basé sur la version 4.5 d'OS/2. Jusqu'en 2011 les versions se succéderont avec parfois 5 ans d'attente. eComStation ajoutera de nombreuses fonctionnalités à OS/2, comme le support de l'ACPI, les partitions JFS bootables ainsi que le port de certains logiciels libres comme OpenOffice, Firefox, VLC, Scribus, Samba, Qt 3 et 4, CUPS,

GCC et d'autres. À l'heure actuelle, la version 2.2 est toujours annoncée mais le site web semble bien mort. En 2006 IBM a cessé officiellement tout support d'OS/2.

Installation d'OS/2 Warp 4

OS/2 Warp peut s'installer sur des machines physiques ou virtuelles. Dans ce dernier cas, les réglages peuvent être assez pointus et je vous conseille de partir sur la version 4.52 et non sur la 4.00.

Choix du matériel

OS/2 Warp 4 est un système d'exploitation 32 bits pour processeurs x86. Une version pour PowerPC a existé mais là le choix du matériel est vite fait : IBM Power Series 830/850 desktops uniquement. Si vous possédez ce type de matériel prévenez-moi, je serais extrêmement heureux de tester :) La configuration minimale est la suivante :

- Processeur de type Pentium MMX ou supérieur ;
- 16 Mo de RAM ou plus (4Go maximum, OS/2 Warp 4 ne supportant pas le mode PAE) ;
- Disque dur de 2Go ou plus. Attention, si vous utilisez le driver IBM, celui-ci ne sait pas reconnaître plus de 8Go ;
- Écran VGA ou plus ;
- Clavier et souris PS/2 pas USB, du moins pour l'installation ;
- Carte réseau plus que fortement recommandée ;
- Lecteur de CD-ROM IDE ou SCSI ;
- Lecteur de disquettes 3¹/₂ HD obligatoire.

Afin de voir si votre matériel est supporté, je vous conseille d'aller faire un tour sur le site <https://hobbes.nmsu.edu/?path=%2Fpub%2Fos2%2Fsystem%2Fdrivers> qui recense 95 % des drivers disponibles pour OS/2. Cependant, l'installateur ne permet pas d'installer un driver spécifique en cours de route (enfin, pas sans bidouiller), donc choisissez du matériel qui était standard en 1995, c'est à dire disque IDE ou SCSI, carte réseau 3C509, NE2000, Intel EtherExpress ou IBM Token Ring, carte VGA PCI ou ISA avec un chip Matrox, S3 ou Cirrus Logic.

Si vous voulez installer OS/2 sur une machine virtuelle de type VirtualBox, faites très attention aux réglages fins du genre ACPI, VTx et autres. Je vous conseille plus que fortement avec VirtualBox de choisir le driver VGA tout court lors de l'installation, le SVGA

risque de faire planter la VM et il s'agit d'une option complexe à modifier après.

Téléchargement et installation d'OS/2

ATTENTION : OS/2 n'est ni libre ni gratuit, même encore aujourd'hui. De nombreux sites proposent de télécharger des ISOs des diverses versions d'OS/2 (archive.org par exemple) mais vous n'êtes pas censé les installer. Ceci dit, pour faire quelques tests dans un cadre purement académique, je ne pense pas qu'IBM vous enverra ses juristes. En tous cas, moi, je ne les ai pas vus :)

Note préliminaire : dans ce qui suit, les logiciels et commandes utilisés pour créer les disques et disquettes sont spécifiques à Linux. Si vous utilisez un autre OS, je vous laisse faire la conversion.

Il existe deux versions sur archive.org d'OS/2 Warp 4. Une version 4.0 qu'il faudra impérativement mettre à jour pour pouvoir faire quoi que ce soit avec et une version 4.52 déjà au top de l'époque. La première nécessitera une installation "à la dure" à grand renfort de disquettes et de fixpacks gravés sur un CD. Si vous décidez d'utiliser celle-ci, voici quelques conseils :

- Lorsque vous téléchargerez l'archive, vérifiez bien que vous avez quatre fichiers :
- Une ISO
- TROIS images de disquettes : Une de boot et deux d'installation. Il est fréquent que la disquette de boot ne soit pas présente dans l'archive, ce qui en limite fortement l'intérêt...
- Pour créer le CD, procédez comme d'habitude avec un logiciel de gravure type K3B ou autres.

Pour les disquettes, tout dépend de ce que vous avez trouvé. Soit vous avez eu de la chance et le format est simplement une image physique de la disquette, soit vous avez récupéré un format dit IBM SaveDskF (extension .DSK), et là, ça se complique un peu. Dans le premier cas, la commande suivante fera l'affaire :

```
dd if=<nom_de_l'image> of=/dev/fd0 bs=512
```

Dans le cas contraire, vous pouvez récupérer un utilitaire à l'adresse suivante :

<https://ecsoft2.org/loadskf-savedskf>. Cet outil

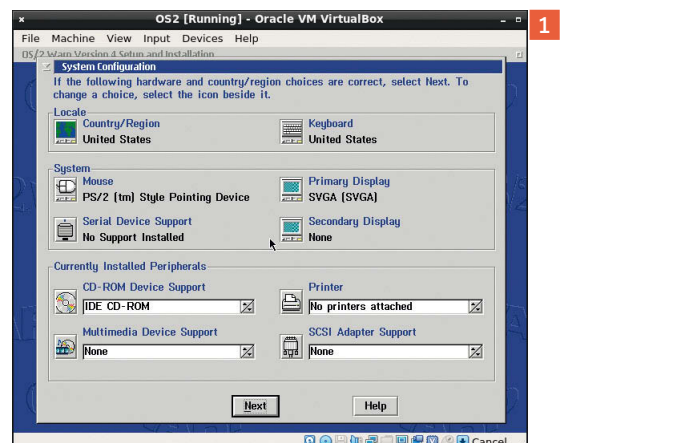
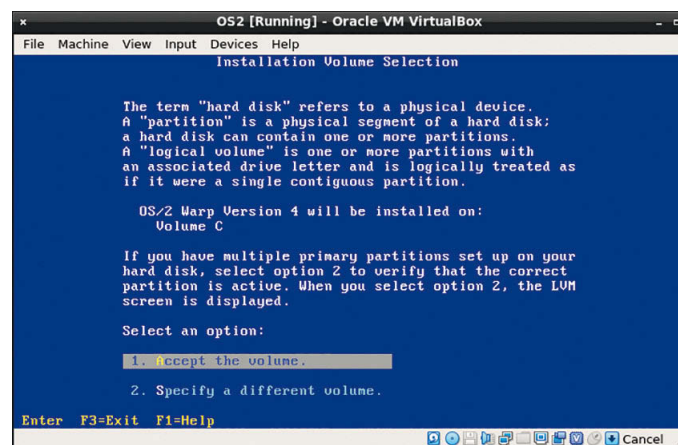
ne fonctionne que sous DOS et sous OS/2. La meilleure des options est donc d'installer vite fait un MS-DOS sur la future machine OS/2 le temps de créer les disquettes d'installation.

En ce qui concerne les disquettes 2 et 3,

je vous conseille fortement de ne pas utiliser celles fournies avec l'image, mais plutôt d'aller récupérer celles créées gentiment par Elmar Stellnberger ici : <https://www.elstel.org/OS2Warp/InstallUpdate.html>. En effet, elles intègrent directement le driver disque appelé Daniela qui permet d'accéder aux disques récents sans problème.

Si vous décidez d'utiliser directement la version 4.52, elle est fournie sous la forme de deux ISO et c'est la deuxième qui est bootable. Ensuite il vous faudra bien examiner chaque écran d'installation et choisir les drivers correspondant à votre matériel. En cas de doute, choisissez le plus générique possible, il sera toujours temps d'installer ensuite un driver idoine téléchargé sur le site de hobbes ou sur le ftp d'IBM.

L'installation à partir des CD se fait nettement plus simplement. On insère le deuxième CD et on boot dessus, puis on suit les instructions à l'écran. Pour installer le boot manager d'OS/2, il faut choisir l'option "Specify a different volume" lors de l'installation et le système nous donne la main sur le partitionnement. **1**



Quelques recommandations au sujet de ce mode d'installation :

- Mettre un disque dur pas trop gros (un 40 Go passe sans problème) sur lequel on ne compte pas conserver de données ;
- Au premier boot, l'installateur va créer d'office un volume prenant tout le disque et rebooter, ne pas s'inquiéter ;
- Prendre l'option "formatage rapide" ou prévoir de lui laisser la nuit. Compter environ huit heures pour un formatage lent d'un disque de 40Go !
- Avoir à portée de main les drivers réseau, sur disquettes de préférence. La modification est faisable ultérieurement, et dans ce cas, ne pas installer de réseau du tout sinon vous risquez d'être bon pour aller trifouiller à la main dans le CONFIG.SYS ;
- En ce qui concerne le driver écran, prendre VGA de base sauf si votre carte est reconnue. Ensuite, il sera temps de modifier cela et d'installer soit les drivers officiels, soit l'excellent SciTech SNAP Graphics disponible sur hobbes ; **2**

Maintenant, la première application à installer selon moi est le gestionnaire de paquets d'Arcanoe disponible ici :

<https://ecsoft2.org/anpm-arca-noae-package-manager>

C'est un simple exécutable à lancer. Il vous permettra d'installer tout un tas de logiciels de manière simple. Il est basé sur un port OS/2 du couple yum/rpm bien connu des utilisateurs de Linux Redhat et dérivés. De plus, ce logiciel est open source sous licence GPLv3, ce qui ne gâche rien, bien au contraire.

Que peut-on faire aujourd'hui avec OS/2 ?

OS/2 semble appartenir au passé. C'est une très belle pièce d'histoire mais est-ce seulement cela ? Il semblerait bien que non. En effet, malgré son échec commercial auprès du grand public, de nombreuses entreprises l'ont adopté et l'ont utilisé pour plein d'autres choses que pour monter un serveur ou pour mettre sur des postes de travail. Sa relative légèreté, son multi-tâches natif et sa stabilité lui ont valu de se voir utilisé aussi comme solution industrielle, par exemple pour gérer les cartes d'accès au métro de New York ou pour faire fonctionner des distributeurs de billets, mais aussi pour piloter des machines industrielles à plusieurs centaines de milliers de dollars. À l'heure actuelle ces dernières sont encore

en production et OS/2 reste le seul moyen de les faire fonctionner. Cependant, les ordinateurs vieillissent et ne sont plus réparables par manque de pièces détachées et OS/2 Warp ne peut pas s'installer sur du matériel moderne (UEFI, USB, etc.). Une clientèle – de niche certes mais de grande niche – existe donc. C'est le constat qu'ont fait en 2014, de nombreux développeurs d'eComStation. Ils décident de ce fait de fonder une société nommée Arca Noae pour reprendre le développement sur de bonnes bases. Cette initiative a donné le jour à ArcaOS qui reprend le flambeau.

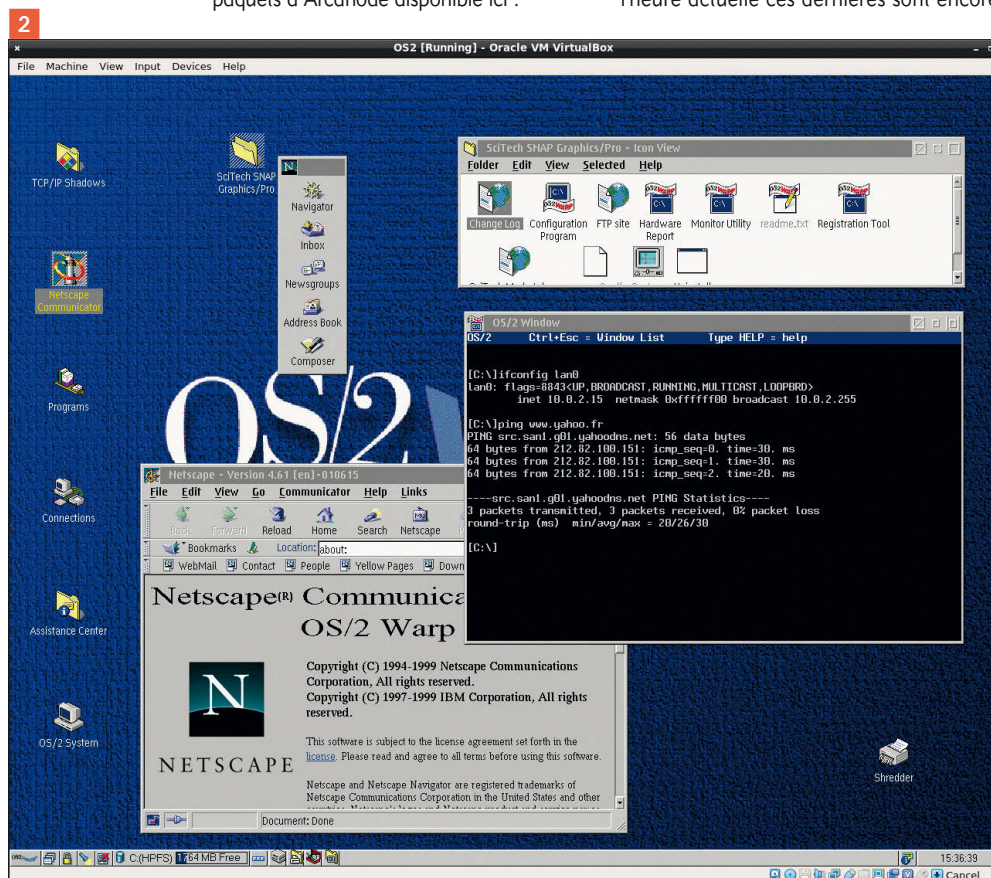
J'ai eu l'occasion d'échanger avec Lewis G. Rosenthal, le "Managing Member" d'Arca Noae qui a gentiment bien voulu répondre à mes questions.

CP : pouvez-vous nous dire quel genre de société utilise encore OS/2 ?

LR : Arca Noae effectue un gros travail de consultant sur les systèmes OS/2 de grosses entreprises. Nos principaux clients sont plutôt dans les secteurs de la finance, de l'industrie ou de la fabrication. Nous avons aussi des clients dans les métiers de la vente de matériel technologique (grossistes) chez qui OS/2 est encore utilisé pour gérer les inventaires, les listes de prix et les commandes.

Nous avons aussi travaillé pour des sociétés appartenant au groupe des "Fortune 500 companies" dans lesquelles OS/2 est encore utilisé pour des applications critiques. Ces applications ont été écrites dans les années 1990, mises à jour au début des années 2000 et depuis, soit ces logiciels nécessitent d'être améliorés, soit l'infrastructure sur laquelle ils tournent a tellement changé qu'il n'est plus possible de continuer d'y faire fonctionner des applications basées uniquement sur NetBIOS. Pour ces cas de figure, nous effectuons le portage vers Samba (nous disposons d'un client Samba tenu à jour et d'un serveur Samba en 3.6 actuellement). De plus, nous avons eu des demandes pour mettre en place l'authentification sur de l'Active Directory et nous avons porté la version Kerberos d'Heimdal pour ce faire.

Dans le domaine de la fabrication industrielle, nous rencontrons des systèmes OS/2 en production qui font fonctionner des machines d'assemblage comme des robots ou



des machines de fabrication de circuits imprimés.

Il existe aussi quelques PME qui utilisent encore OS/2 de nos jours. Toutefois nous aimerions bien sûr voir ce marché progresser alors que de plus en plus de préoccupations vis-à-vis de la sécurité et de la vie privée se font jour concernant des systèmes propriétaires plus connus et que nous parvenons à prouver nos compétences sur du matériel moderne. Tout le monde n'a pas besoin d'un système 64 bits pour faire de la comptabilité, imprimer des brouillons et saisir quelques informations en base (ou faire du rétro-gaming).

CP : pourquoi ne sont-ils pas passés sous Windows, Linux, FreeBSD ou autres ?

LR : généralement le problème a été économique. Initialement ces sociétés ont investi énormément d'argent (souvent sur la suggestion d'IBM et souvent avec IBM directement) pour développer des logiciels métier correspondant à leurs besoins. Ensuite IBM leur a conseillé de migrer d'OS/2 vers Linux ou vers Windows. Malheureusement pour beaucoup d'entre eux, le coût du portage d'applications - qui autrement restent parfaitement adaptées à leurs besoins - vers une autre plateforme était nettement plus élevé que celui nécessaire à la maintenance opérationnelle de l'infrastructure OS/2. Du coup, des sous-réseaux OS/2 ont été conservés, totalement intégrés dans une infrastructure hétérogène, coexistant avec Windows et Linux.

Parfois, face à un problème d'incompatibilité matérielle, OS/2 a été virtualisé sous VirtualPC ou VirtualBox sur des machines Windows. Du fait que, comparés aux standards actuels, OS/2 ne nécessite que très peu de ressources (c'est un OS 32 bits après tout), il peut fonctionner dans une "boîte" sans provoquer une baisse de performance perceptible. Cependant, la mission d'Arca Noae est de conserver OS/2 sur le "métal", autant que faire se peut, afin de lui permettre d'exprimer tout son potentiel sur du matériel moderne. C'est là qu'ArcaOS sort du lot avec le support du SMP, de JFS et (maintenant) de PAE permettant d'utiliser la mémoire au-delà des 4Go comme RAM disk.

CP : quels sont les besoins exprimés par vos clients qui vous poussent à porter

OS/2 sur du matériel récent (vous m'aviez parlé d'UEFI) ?

LR : le support d'UEFI est encore en cours de développement. À l'heure actuelle nous pouvons démarrer et installer ArcaOS sur des systèmes UEFI sans mode de compatibilité. Il nous reste encore pas mal de travail à effectuer avant de pouvoir dire que nous avons totalement intégré le support de l'UEFI dans l'installateur et avant d'avoir un support complet de la vidéo pour le DOS et les applications Windows. Cependant ArcaOS est probablement le seul OS encore maintenu qui possède un support natif des applications DOS et Windows 3.1. Je crois me souvenir que, pas plus tard qu'il y a quelques années, l'aéroport d'Orly utilisait encore des applications Windows 3.1 pour le contrôle aérien. Avec ArcaOS, ils auraient pu conserver le système et le faire tourner sur du matériel récent, peut-être sans aucune modification du code.

Notre support de l'UEFI est dicté par le marché. En effet Intel a annoncé la fin du mode de compatibilité sur ses chipsets pour la fin de l'année. Cela signifie que la dernière génération de matériel serait inutilisable avec un système se basant sur le BIOS traditionnel. Bien sûr, ça ne pose pas de problème pour tout le matériel plus ancien déjà en place mais nous pensons qu'il est important pour nous d'être capables d'installer et de faire fonctionner nos produits sur du matériel très récent et de maîtriser le processus de démarrage de ces machines. C'est pour cela que nous investissons lourdement dans la R&D afin d'arriver à ce résultat.

GPT est un autre domaine proche de celui-ci. Pour le moment, OS/2 (et ArcaOS en particulier) ne supporte que des disques de 2To ou moins, avec le partitionnement traditionnel de type MBR. Ne pas être en mesure d'utiliser les récents SSD ou disques durs de grande capacité, ni de pouvoir faire un multi-boot sur un disque partitionné GPT, nous pose un vrai problème. C'est pourquoi GPT sera la prochaine étape dans l'évolution d'ArcaOS. C'est juste une question de nous garder en phase avec le matériel disponible actuellement.

Le support de l'USB 3 (contrôleurs xHCI) est en approche finale et notre pile USB existante est maintenant presque entièrement 32 bits. Ceci offre un contraste saisissant avec la pile USB d'IBM dans Warp 4 qui s'est arrêté à l'USB2 (EHCI) en étant uni-

quement en 16 bits. Cette avancée nous permet d'utiliser les plus récents systèmes et les périphériques USB Super Speed pour les sauvegardes ou autres.

Pour en revenir à votre question, nos clients nous ont exprimé leur besoin d'un support des machines modernes et d'une coexistence avec d'autres OS supportant l'UEFI. Ils ont besoin de pouvoir brancher un clavier ou une souris en USB sans se demander s'il s'agit d'un port connecté à un contrôleur EHCI ou xHCI.

Le WiFi est un autre domaine qui nécessite des améliorations et le support de plus d'adaptateurs sans fil est dans notre planning. Nous fournissons déjà un excellent support des réseaux câblés grâce à nos pilotes MultiMac qui ont été principalement portés depuis FreeBSD ("portés" n'est franchement pas le bon terme ; en fait MultiMac fournit un environnement natif OS/2 dans lequel le code FreeBSD peut être déposé directement et fonctionner sans aucune modification). MultiMac sera étendu aux drivers WiFi essentiellement de la même façon.

Dans notre planning, nous avons essayé de mettre en priorité le boot, puis la connexion des périphériques locaux et finalement le réseau. C'est fondamentalement ce dont nos clients ont besoin, et dans cet ordre. Bien entendu, nous sommes aussi disposés à développer des drivers pour des applications spécifiques. Un de nos récents contrats dans le secteur manufacturier concerne le développement d'un driver pour une carte réseau ARCNET. Comme pour nombre de nos contrats, ce fut pour permettre le fonctionnement d'une application métier sur du matériel récent pour lequel aucun pilote n'était disponible de la part du fabricant.

CP : est-ce difficile d'adapter OS/2 aux nouveaux matériels et aux nouvelles architectures ? Prévoyez-vous de faire tourner ArcaOS uniquement sur x86 ? ArcaOS sur une Raptor ou un IBM Power 9 serait super ;)

LR : de par la nature de notre contrat de licence avec IBM, une partie de nos développements est limitée par le code sous-jacent d'OS/2 Warp 4.52. Nous avons remplacé le boot loader et nous avons écrit un nouveau composant de dump mémoire pour le noyau. Mais ceci dit, le noyau en

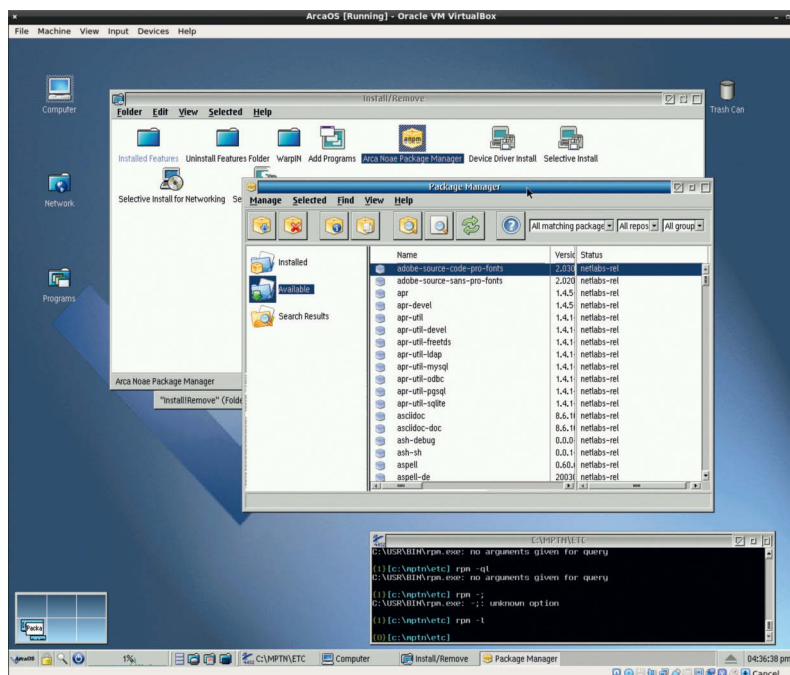
lui-même reste inchangé. C'est du code x86 uniquement donc il est extrêmement peu probable de voir un jour apparaître un noyau 64 bits pour ArcaOS (sans parler d'applications 64 bits à faire fonctionner dessus). Il en est de même pour une version ARM ou autre d'ArcaOS. ArcaOS est un OS/2 32 bits pour x86 nettement amélioré (il supporte nativement jusqu'à 64 processeurs/cœurs) :-)

CP : de votre point de vue, quel sera le futur d'ArcaOS ?

LR : nous voulons faire un grand pas vers le marché du rétro-gaming avec ArcaOS 5.1 (ou 5.2). Il semble y avoir une énorme demande pour une solution capable de faire fonctionner des jeux vidéo classiques pour MS-DOS, Windows 3.1 ou OS/2 et nous sommes les seuls capables de réaliser cela sans toutes ces couches d'émulation qui peuvent ralentir les jeux sur d'autres plateformes. Notre driver vidéo SNAP devrait être amélioré pour supporter plus de puces graphiques et fournir une accélération vidéo à ces applications.

Nous désirons aussi explorer le marché de l'informatique mobile sur des plateformes plus classiques (ordinateurs portables). Nous les voyons comme des remplaçants potentiels des postes de travail là où ce n'est pas déjà fait pour certaines applications ne demandant pas de systèmes 64 bits. Et surtout là où la sécurité et le respect de la vie privée sont primordiaux (ArcaOS ne "téléphone" jamais à la maison, n'envoie jamais d'informations d'aucune sorte à qui que ce soit).

Pour les curieux voilà ce que donne ArcaOS à l'heure actuelle : **3**



En conclusion

OS/2 est donc plus vivant que jamais et son développement se poursuit, dans un cadre certes assez restreint, mais correspondant à la demande du marché.

J'aurai sans doute l'occasion de faire un essai complet d'ArcaOS dans un prochain article pour vous en dire plus.

Pour le moment une communauté de fans porte de nombreux logiciels libres vers OS/2 et maintient des sites web fort intéressants.

Voici une liste non exhaustive de liens qui méritent le détour si vous désirez vous lancer dans l'aventure :

<https://www.elstel.org/OS2Warp/> (Installation d'OS/2)

<https://www.tavi.co.uk/os2pages/> (Informations diverses et utilitaires)

http://www.edm2.com/index.php/Main_Page (Une vraie mine d'or)

<https://hobbes.nmsu.edu/> (LE site incontournable)

<http://www.abiyo.net/os2/> (Pour les problèmes réseau)

<https://os2ports.smedley.id.au/> (Les portages Open Source vers OS/2)

<http://www.2rosenthals.net/wordpress/tag/os2/> (Le blog du patron d'ArcaOS)

MESSAGE PRIORITAIRE :
Programmez! n°241, prochain numéro
date stellaire : 5 juin 2020

Développeur freelance : comment le devenir ?
comment le rester ?

Dossier spécial Red Hat Openshift



Marie-Lorraine CHAMLA
Conseil en gestion de projet web
et marketing digital
MLC Conseil



Antoine GILLES
Responsable Marketing
et Digital
Groupe ENI

L'Éco-conception web : une nécessité

La rédaction : cet article fait suite à notre dossier éco-conception publié dans *Programmez!* 239.

Aujourd'hui, plus que jamais, notre société fait face à des enjeux inédits avec en tête le dérèglement climatique. L'activité humaine a un impact fort sur notre planète et il est du devoir de chacun d'en prendre conscience et d'essayer, à son niveau, de limiter son empreinte environnementale.

Lorsque l'on travaille dans le numérique, il est facile de penser que l'on n'est pas concerné! Grosse erreur, **l'impact du numérique** sur l'environnement est bien réel et représente **4 % des émissions de gaz à effet de serre dans le monde, et au rythme actuel : dépassera les 9 % en 2030 !**

Refroidissement des serveurs, production du hardware (ce qui induit l'extraction de terres rares, la consommation électrique basée sur le charbon pour le fabriquer en Chine et l'acheminer en Europe), consommation électrique des réseaux, etc. L'empreinte écologique des sites web est grandissante et cela s'explique notamment par le poids des pages qui a été multiplié par 6 ces dernières années! En tant que développeur et concepteur de site web, il est tout à fait possible d'allier performance et sobriété! Et c'est tout le sens de l'écoconception justement!

En bref, **l'écoconception numérique** revient à « concevoir des outils numériques plus fonctionnels, plus rapides, consommant moins de ressources, des objets connectés plus robustes, réparables, générant moins de déchets : l'ensemble de ces notions peuvent et doivent être étudiées dès la phase de conception afin de concilier transition digitale et transition environnementale. C'est ce que propose l'écoconception. » ([Source Green IT](#))

Pour vous aider, voici **les 10 bonnes pratiques** à appliquer pour créer un site web éco-responsable :

1 Les vidéos

Elles représentent plus de 75 % des flux Internet et ce chiffre devrait s'élever à 84 % d'ici la fin de l'année 2020. La vidéo a un impact environnemental non négligeable.

Avant toute chose, il est important de se poser la question de la pertinence de mettre des vidéos dans les bannières et sur les différentes pages d'un site. Est-ce que cette vidéo est indispensable? Une image ne ferait-elle pas l'affaire? Avez-vous pensé au podcast? Ce format audio est beaucoup moins énergivore que la vidéo, mais tout aussi impactant.

2 Ré-encoder les vidéos

Si elles sont absolument indispensables sur un site, on peut les ré-encoder avec des outils gratuits et open source comme [Handbrake](#) qui permet de compresser ses vidéos tout en gardant une qualité acceptable. Pour vous donner un exemple réalisé par The Shift Project :

- avant traitement, voici les caractéristiques de la vidéo de test originale : format 1920×1080, 9mn13, mp4 = 132,90 mo
- après traitement : le gain en poids est de 57,83 mo soit 44 % de réduction! La vidéo au format de 1920×1080, 9mn13, mp4 ne pèse plus que 75,07 mo.

Cette disposition permettra également à votre site web d'être plus rapide et ainsi mieux référencé sur les moteurs de recherche. Une action gagnante sur tous les plans!

3 Téléchargement vs streaming

Le rapport « [L'insoutenable usage de la vidéo en ligne](#) » publiée par The Shift Project indique que le visionnage de vidéos en ligne aurait produit « 300 millions de tonnes de CO2 en 2018 ». Ces propos sont cependant à nuancer, car l'impact du streaming dépend de nombreux paramètres tels que la taille de l'écran

(smartphone, tablette, TV 4K, écran PC, etc.), la résolution de la vidéo ou encore le type de connexion. Il semble donc plus responsable de proposer ses vidéos en téléchargement plutôt qu'en streaming. Si malgré tout, vous optez pour cette dernière option, programmez par défaut la lecture en basse définition.

4 Visuels et photos

De la même manière que la vidéo, mais dans une moindre mesure, il convient de limiter au maximum les visuels et photos d'illustration choisis. Leur poids, leur taille doivent être réduits au strict nécessaire afin de minimiser l'impact du site web tout en assurant un contenu cohérent. Rappelons que la consommation énergétique d'un site web est en rapport direct avec son poids multiplié par le nombre d'utilisateurs ou de visiteurs qu'il rassemble.

5 Contenu

Quelles sont les informations indispensables à présenter sur mon site web pour atteindre mon objectif? Première étape : définir l'objectif de son site. Puis, en fonction de l'objectif choisi, définir les informations clés à valoriser sur le site. Il faudra faire la part des choses entre le contenu « essentiel » et le contenu « secondaire » pour être pertinent sans se perdre dans des détails qui n'auront pas de valeur ajoutée. Il ne faut pas hésiter également à supprimer les contenus devenus obsolètes. Car même si pour des raisons de référencement, on souhaite souvent les conserver, s'ils sont obsolètes, le message envoyé aux visiteurs sera contre-productif.

6 Algorithmes

Nous concevons et utilisons des algorithmes pour de nombreux usages :

recommandation de produits pour le e-commerce, suggestions de contenus à lire ou visionner pour les plateformes, suggestions de profils sur les sites de rencontre, etc. Dans une logique d'écoconception, ceux-ci ont aussi un impact important sur la consommation de ressources. En effet, au lieu de proposer toujours plus de contenus, l'algorithme peut aussi proposer seulement des contenus de plus grande qualité, un contact avec un opérateur, voire ne pas faire de suggestions supplémentaires, si on détermine qu'elles ne seraient que peu pertinentes.

Ces choix auront des conséquences non négligeables sur l'usage fait par les visiteurs et utilisateurs des applications ou sites web, qui « consommeront » moins (y compris d'énergie), mais aussi auront une meilleure perception de la qualité des contenus proposés : proposer moins, mais mieux.

7 CSS

Là où on voit encore sur certains sites, des CSS différents utilisés pour plusieurs pages aux contenus proches. S'interroger sur la pertinence de modifications mineures dans la présentation des contenus peut permettre de conserver les mêmes CSS pour l'ensemble d'un site. Et donc de diminuer aussi la consommation énergétique de ce dernier.

8 Outils tiers

Limiting the number of external tools (Analytics, ads, retargeting, social networks, etc.) is to be considered. Is your activity

nécessite d’être soutenue par de la publicité ? Est-ce qu’il est indispensable pour vous de mettre en avant vos réseaux sociaux sur votre site ? Il est important de se poser ces questions afin de prendre des décisions en conscience. L’idée étant de se limiter au strict nécessaire : par exemple, le formulaire de contact ou le CA généré restent les meilleurs indicateurs de performance pour les sites commerciaux.

9 Outils de mesure

Il existe des outils qui permettent de mesurer le poids et donc les performances de son application ou de son site web. Certaines solutions permettent aux développeurs de **coder de manière plus durable**, c'est à dire en optimisant les besoins en énergie : plus ou moins intégrés aux outils de développement et de tests. Ils permettent aux développeurs de qualifier et d'optimiser en direct la qualité durable d'un code informatique via un audit statique et des mesures en temps réel. Citons par exemple [Greenspector](#), [Ecometer](#) ou encore [EcolIndex](#).

10 Hébergement green

Tout le contenu de votre site web est hébergé sur des serveurs, dont la majorité sont hébergée dans des centres de données. Ces énormes installations hébergent de nombreux serveurs, et ce pour tous types de sites web. Ces serveurs doivent continuer à fonctionner dans des environnements refroidis et contrôlés. Ils nécessitent donc une grande quantité d'énergie. On comprend

aisément l'impact de l'hébergement sur l'environnement.

Choisir un hébergeur « vert » alliant faible consommation électrique et énergies renouvelables, voire réutilisation de la chaleur fatale des serveurs en économie circulaire est une bonne option.


Plusieurs solutions existent, par exemple : <https://www.opportunités-numériques.com/hebergement-ecologique/>

Point important : les **CER** (certificat d'énergie renouvelable) sont générés par des entreprises spécialisées dans la création d'énergies renouvelables. En les achetant, un hébergeur web peut attester qu'il a contribué à la production d'une certaine quantité d'énergie renouvelable. Argument intéressant pour faire votre choix.

Vous voulez aller plus loin dans votre démarche d'écoconception ? Nous vous conseillons plusieurs ouvrages et ressources pour approfondir le sujet :

<https://communication-responsible.ademe.fr/sites/default/files/les-115-bonnes-pratiques-ecoconception-web-frederic-bordage.pdf>
<https://www.greenit.fr/2020/03/10/ecoconception-numerique-un-guide-de-45-bonnes-pratiques/>
<https://www.greencode-label.org/>

Une chose est sûre, vous pouvez d'ores et déjà vous féliciter pour votre démarche ! L'écoconception de site web est salutaire et permet de prouver que le numérique peut aller de paire avec durabilité !



1 an de Programmez!

ABONNEMENT PDF :

35 €

Abonnez-vous sur :

www.programmez.com

Tests : encore et toujours



*Que serait une saison de **Programmez!** sans un dossier sur les tests ? Nous n'osons même pas l'imaginer. Le test, on ne devrait même pas y penser. Il devrait être un réflexe naturel dans chaque projet. On éviterait de nombreuses déconvenues.*

Le test est là pour nous aider quand il est pensé et intégré à la chaîne de développement. Il n'y a rien de pire que des tests rajoutés en urgence. Il faut en finir avec le mode pompier !

Finalement, le test est comme une cuisine : il faut suivre la recette...

Avec les nouvelles architectures, la complexité toujours croissante des plateformes et des apps, le test doit évoluer et suivre le mouvement. Mais il faut que les testeurs et les développeurs suivent eux aussi le mouvement.



Marc Hage Chahine
Key member centre d'expertise
de test d'Altran (ITQ)
Créateur et animateur du blog
La taverne du testeur



Petite introduction à la cuisine des tests

Les tests, et plus précisément les campagnes de tests, c'est comme la cuisine... Et plus exactement comme un repas ! Vous n'en n'êtes pas encore convaincu ? Vous le serez après avoir lu cet article ! Bon appétit.

Repas et campagnes de test

Je parle régulièrement (dans mon livre « Tout sur le test logiciel » et divers articles de la taverne du testeur) des 3 types de campagnes de test suivantes :

- **Les campagnes de tests vitaux** (tests fumigatoires selon ISTQB), à faire à chaque commit pour s'assurer l'absence d'anomalie critique ;
- **Les campagnes de régression**, à faire en fin de validation de fonctionnalité pour assurer l'absence d'anomalie majeure ;
- **Les campagnes de validation** (test d'évolutions), à faire pour valider toute nouvelle fonctionnalité.

Ces campagnes ont différentes caractéristiques :

- Elles ont des buts différents ;
- Elles sont effectuées plus ou moins souvent ;
- Elles ont un nombre de tests différents.

C'est le choix de tels critères qui permet de définir des typologies de campagnes de tests. A telle enseigne que James Bach (co-auteur des test exploratoires avec Cem Kaner) propose une heuristique pour créer son approche stratégique des tests d'un produit à partir de l'environnement du projet, des critères liés à la qualité attendue, des éléments du produit à tester pour en déduire les techniques de tests à appliquer. Pour rester sur des cas simples que l'on rencontre dans la vie de tous les jours nous allons faire l'analogie avec les repas dont on peut distinguer 3 types :

- Les repas quotidiens ;
- Les repas « du dimanche », au restaurant ou avec des amis ;
- Les repas de fête.

Vous l'avez sûrement déjà compris, nous pouvons déjà faire un premier parallèle :

- **Les campagnes de tests vitaux sont l'équivalent des repas quotidiens.** En effet, ils sont effectués très fréquemment, et ont pour but d'assurer la subsistance de



© dth84

l'application (ou de la personne) et ne sont pas forcément sophistiqués ou variés.

- **Les campagnes de tests de régression sont plus proches des repas du dimanche.** Moins fréquents que les repas quotidiens ces derniers n'en sont pas moins réguliers. Ils sont plus complets et plus riches. Le but ici est d'aller plus loin qu'une simple subsistance et de partager de meilleurs moments avec les siens, mais le problème est qu'ils peuvent ressembler au sempiternel « rosbeef du dimanche » avec des tests de régressions qui évoluent peu jusqu'à devenir insipides (l'ISTQB nomme cette lente désuétude des tests le « paradoxe du pesticide » à l'image des bugs deviennent progressivement insensibles à un même insecticide).
- Enfin, **les campagnes de validation sont l'équivalent des repas de fête.** En effet, ces campagnes tout comme ces repas sont rares, souvent coûteux, copieux et variés et le but ici n'est plus une subsistance mais bien un plaisir pour le repas ou

une recherche approfondie pour la campagne de validation.

L'analogie ne s'arrête pas là. La cuisine est bien plus variée que la simple notion de « repas ».

Repas, plats et campagne de test

En effet les repas sont composés de différents plats et de boissons. De même il existe un grand nombre de boissons, et pour les plats, il y a de nombreux ingrédients !

La notion des différents plats dans un repas est très importante dans la gastronomie et plus généralement dans les repas. On peut décrire le processus de l'ensemble des plats d'un repas de cette manière (chaque plat n'étant pas obligatoire pour un repas et il est évidemment possible d'en ajouter) :

- L'apéritif ;
- L'entrée ;
- Le plat principal ;
- Le fromage ;
- Le dessert.

On imagine généralement :

- Un repas quotidien ayant 2 plats (entrée/plat ou plat/dessert) ;
- Les repas entre amis l'ensemble des plats avec ou sans le fromage ;
- Les repas de fête avec l'ensemble des plats, voire certains qui sont doublés. Il est également fréquent de trouver, lors de ces repas, des spécialités que l'on ne déguste généralement que lors de ces repas comme le chapon ou la dinde à Noël.

Ces phases du repas se retrouvent également dans les campagnes de tests. En effet, les tests ne se font pas tous en même temps. Il est également possible d'avoir des livraisons échelonnées afin de valider progressivement le logiciel. Comme pour les repas, les campagnes de tests n'ont pas besoin du même nombre d'étapes pour être exécutées. On peut imaginer par exemple pour les différentes campagnes :

- Tests vitaux : tous les tests unitaires et des tests fonctionnels ;
- Tests de régression : tous les tests unitaires, tous les tests fonctionnels automatisés, des tests de performances et les tests d'intégration ;
- Tests de validation : tous les tests unitaires, tous les tests fonctionnels automatisés, des tests exploratoires sur la fonctionnalité à valider, des tests de performances, les tests d'intégration et les tests d'acceptation (avec une bêta test). A l'image des spécialités des repas de test, une grande partie des tests de validation ne sont dégustés que lors de ces campagnes !

On peut également penser aux différentes activités de test selon l'ISTQB. Ces différentes activités sont :

- La planification des tests ;
- L'analyse des tests (ce qu'il faut tester) ;
- La conception des tests ;
- L'implémentation des tests ;
- L'exécution des tests ;
- L'évaluation des critères de sortie ;
- La clôture des tests.

Penchons-nous maintenant sur un nouveau parallèle : les ingrédients et types de boissons qui se révèlent être les tests des campagnes !

Composition des plats et des campagnes de tests

Les ingrédients et boissons représentent d'ailleurs très bien les tests. En effet, pour faire un plat (ou un repas) il faut :

- Une (ou plusieurs) recette(s) ;
- Différents ingrédients (féculents, légumes, viandes, épices, liquides et potentiellement boisson) ;
- Suivre un processus ;
- Finir dans le temps imparti : finir de cuisiner le dimanche midi alors que les invités arrivent le samedi soir n'est pas une option !).

Pour une campagne de test, c'est pareil !

- Notre recette devient notre plan de test ;
- Les ingrédients, les différents types de tests à exécuter ;
- Le processus : l'ordre d'exécution et la manière d'exécuter les tests (automatisé, manuel...) ;
- Finir dans le temps imparti étant tout simplement la date de fin de campagne.

De même, dans un repas constitué de plusieurs plats comme l'entrée, le plat principal, le fromage et le dessert, il est fréquent de prévoir à l'avance le temps passé à déguster chaque plat afin de servir au moment opportun les plats suivants !

Cette stratégie d'optimisation de la dégustation se retrouve également dans les campagnes de tests et peut s'apparenter à une planification des tests faite à partir d'une stratégie définie et étudiée à l'avance.

Ingrédients et tests

Nous arrivons à la partie que je trouve la plus intéressante de l'analogie, la comparaison entre tous les ingrédients disponibles en cuisine avec l'ensemble des tests à disposition du testeur.

Attention, je vais prendre ici pour cet exemple les 8 facteurs (que j'appelle famille dans mes articles) de l'ISO-25010. Cela pourrait être également fait avec différents types de tests ou les niveaux de test ou même (en s'éloignant du sujet) différentes fonctionnalités d'un logiciel. Tout comme il y a des familles d'ingrédients, il y a aussi des familles/types de tests.

Prenons ces correspondances :

- Les féculents => les tests fonctionnels. Les tests fonctionnels peuvent prendre différentes formes (exactitude, complétude et aptitude à l'usage) tout comme les féculents (pâtes, pomme de terre, pain...).

A noter : je prends ici les féculents pour les tests fonctionnels car les tests fonctionnels sont la base de la plupart des campagnes de test comme les féculents qui sont la base de la plupart des repas (pain, riz, pâtes...) ;

- Les liquides (huile, eau, alcool, sodas, lait...) => les tests de performances ;
- Les protéines (viande, soja, lentilles...) => les tests de compatibilité ;
- Les fruits et légumes (concombres, pommes de terre, carottes, melons, raisins) => les tests d'utilisabilité ;
- Les graines (cacaahuètes, pistaches...) => les tests de fiabilité ;
- Les produits laitiers (lait, fromage, yaourt...) => les tests de sécurité ;
- Les matières grasses (beurre, huile, margarine...) => les tests de maintenabilité ;
- Les épices et autres condiments (cumin, sel, sucre, moutarde...) => les tests de portabilité.

A noter : les plus observateurs d'entre vous auront noté que l'huile se retrouve dans deux catégories car de nombreux tests vérifient en fait plus qu'un unique critère de qualité.

Pour être en bonne santé, les diététiciens recommandent de manger équilibré, ce qui revient à manger régulièrement des ingrédients de toutes les familles. C'est aussi le cas dans le test. Afin d'assurer la bonne qualité d'un logiciel il faut vérifier l'ensemble des critères de qualité !

A noter : comme vous pouvez le voir, certains ingrédients sont dans plusieurs familles. Cela peut également être le cas avec certains tests qui peuvent vérifier plusieurs critères de qualité ou fonctionnalité

De ces informations on peut faire plusieurs déductions :

- Il n'y a quasiment aucun repas (ou même plat) qui ne propose que des ingrédients issus d'une seule de ces familles ! Il en va de même pour les tests, **chaque campagne, même la plus petite, devra inclure plusieurs types de tests**. En effet, ne manger que des féculents entraîne des carences alimentaires avec, par exemple, des carences en protéines ou en vitamines. De cette alimentation résulte une forme de malnutrition qui est dangereuse, à moyen terme, pour notre corps. On peut rappeler les 4 quadrants du test agile de Lisa Crispin qui implique tous les types de tests durant le sprint.
- Plus le repas est important, plus il y aura d'ingrédients différents. En effet, il est tout à fait envisageable lors un repas quotidien de ne manger que des coquillettes avec du jambon (4 ingrédients en comptant le sel et le beurre). Cela est impen-

Tests fonctionnels <ul style="list-style-type: none"> Exactitude Complétude Aptitude à l'usage 	Tests de performance <ul style="list-style-type: none"> Temps de réponse Utilisation de ressource Capacité 	Tests de compatibilité <ul style="list-style-type: none"> Coexistence Interopérabilité 	Tests d'utilisabilité <ul style="list-style-type: none"> Intelligibilité Apprentissage Opérabilité Protection contre les erreurs Ergonomie Accessibilité
Tests de fiabilité <ul style="list-style-type: none"> Maturité Disponibilité Robustesse Récupérabilité 	Tests de sécurité <ul style="list-style-type: none"> Confidentialité Intégrité Rejet (non-rejet) Responsabilité Authenticité 	Tests de maintenabilité <ul style="list-style-type: none"> Modularité Réutilisabilité Analysabilité Modifiabilité Testabilité 	Tests de portabilité <ul style="list-style-type: none"> Adaptabilité Installabilité Remplaçabilité

sable pour un repas de réveillon. Il en est de même avec les campagnes de test ! **Plus la campagne est importante plus il y aura de tests et types de tests exécutés.**

- Un bon plat et un bon repas c'est avant tout un savant mélange du cuisinier qui choisit bien ses ingrédients. C'est pareil pour le test. **Une bonne campagne dépend du choix des tests à exécuter et des objectifs !** De même, le contexte reste important, il n'existe pas de campagne théorique qui serait bonne dans tous les cas.
- Il est impossible de manger tous les ingrédients en un repas tout comme **il est impossible de tout tester**. C'est d'ailleurs un des principes du test. Pour les plus expérimentés, on se rappellera que Per Kroll, l'inventeur de la méthode RUP, à l'aube des méthodes agiles avait déjà fait un parallèle entre la nourriture et les processus de développement en expliquant que l'on ne pouvait pas tout manger mais bien sélectionner ce que l'on allait manger.
- Les ingrédients d'une même famille ne se ressemblent pas forcément, tout comme les tests fonctionnels (ou tout autre type) peuvent être **faits à plusieurs niveaux de génération du code** (tests unitaires, tests d'intégration, tests systèmes).
- Les mêmes ingrédients peuvent avoir différentes formes (pommes de terre sous forme de purée, de frites ou de gratin dauphinois...). De même **un même test peut être manuel, automatisé, semi-automatisé, généré automatiquement...**

- Un bon repas va dépendre des ingrédients, et, selon les saisons, les ingrédients sont plus ou moins bons. **Les bons repas dépendent donc du contexte, ce qui est également le cas des tests !** A ce titre, on peut rappeler le 6ème principe des tests de l'ISTQB mais aussi l'approche de James Bach avec son école du test nommée « Context-Driven Testing »
- A force de toujours manger le même repas on finit par ne plus l'apprécier. C'est l'équivalent du **paradoxe des pesticides** vu plus haut : **plus on exécute les mêmes tests, moins ces tests seront capables de repérer des défauts**. De même, manger toujours le même repas entraîne généralement un autre problème : les carences alimentaires (voir plus bas) !

Outils de cuisine et outils de test

Les ingrédients c'est bien beau, mais ce n'est pas suffisant pour faire de la cuisine. Personne n'est capable de cuisiner sans avoir d'ustensiles (casseroles, couteaux, plats, ...) et de moyens de cuisson (plaques, four, ...) et c'est également le cas pour les tests ! En revanche, une louche n'a jamais fait une bonne recette car le seul outil indispensable pour les tests c'est son cerveau. Il est important de savoir comment on va tester. Suivant ce que l'on doit cuisiner, on prendra tel type de couteau pour désosser une épaule voire une scie pour couper un os. De même, le testeur aura besoin de différents outils pour mettre en œuvre la technique de test utilisée et il pour-

ra recourir à des outils divers et variés. Pas seulement ceux dédiés au test comme les ALM (Application Lifecycle Management – ex : HP ALM, TestLink, XStudio, Refertest...), les outils d'ATDD, les outils d'automatisation, les outils de KDT...

A noter : lorsque vous disposez de personnes qui ont décidé de vous prêter main forte dans la préparation de votre fantastique repas, l'organisation de votre « brigade » dans la conduite des tests s'appelle une « stratégie de test ». Ce terme, souvent utilisé dans tous les sens, signifie ici comment une armée (en grec *stratos*) est conduite (en grec *ageîn*). Ainsi, une stratégie de test décrit comment on testera, mais aussi comment les activités liées au test sont réalisées de façon très formelle dans un document structuré comme le propose l'ISTQB ou par une approche heuristique comme le propose James Bach

Mode de cuisson et manière d'exécuter les tests

En cuisine, les ingrédients et outils sont indispensables, mais il est généralement tout aussi indispensable de gérer la cuisson. Pour cela, il y a plusieurs outils dédiés comme les fours, les cuiseurs vapeur, les plaques de cuisson. Dans de nombreux cas, ces cuissons sont manuelles car on doit les démarrer et suivre l'avancement de cette cuisson en restant derrière. Il existe cependant des moyens « d'automatiser » ces cuissons en programmant son four, lançant son cuiseur vapeur et le laisser tourner pendant

X minutes... Pour les tests c'est pareil !

Sans démarche particulière, les tests sont manuels car le testeur est souvent obligé de s'adapter à chaque situation (ex. un ingrédient ou un ustensile n'est pas disponible, une préparation intermédiaire n'est pas conforme à la recette) mais il existe des moyens plus ou moins poussés pour automatiser des tests. On peut cliquer manuellement pour lancer une campagne qui exécute tous les tests sans avoir besoin d'action du testeur. Cela se rapproche des robots de cuisine qui peuvent réaliser mécaniquement des opérations manuelles.

Il est également possible d'avoir des campagnes qui se lancent automatiquement sans intervention humaine, on est alors ici plus proche de la programmation du four. Que cela soit en cuisine ou dans les tests, que les campagnes soient totalement ou partiellement automatisées, il faut avoir une certaine expérience pour pouvoir automatiser ces tests ! Il faut également bien préparer ces campagnes afin d'avoir les bons tests à exécuter (quel intérêt de lancer un four 2 heures si on a oublié de mettre le plat à cuire dedans, ou pire, si le plat a déjà été cuit ?).

Recette de cuisine et plan de test

En cuisine, il existe de nombreuses recettes. Une recette est une sorte d'algorithme qui permet de réaliser un plat.

On attend plusieurs closes d'une recette parmi lesquelles :

- Une répétabilité : on veut être capable de refaire le même plat à chaque fois.
- Que souhaite-t-on cuisiner ?
- Des actions à effectuer : que faire ?
- Un ordonnancement de ces actions : quand les faire ?
- Les outils à utiliser : comment/avec quoi les faire ?

Dans le test logiciel, il existe un outil similaire que l'on appelle le plan de test. Il répond à différentes questions comme :

- Qu'est-ce que l'on teste ? (Quelle application teste-t-on ?) ;
- Comment on le teste ? (Mise en place, conception...);
- Quels outils utiliser ?
- Quel planning ? (Ordonnancement).

Ce plan de test doit également être répétable, particulièrement pour des campagnes de régression.

On peut d'ailleurs facilement faire un parallèle : voir tableau ci-dessous.

Poussons plus loin la comparaison entre test et nourriture

Les courses : le budget et la planification des tests

Parlons ici des courses que nous devons régulièrement faire pour nous sustenter.

Lorsque l'on souhaite manger chez soi sans

se faire livrer, il est obligatoire d'avoir quelque chose à manger (ou à cuisiner) présent dans son frigo (ou tout autre espace de stockage de nourriture). Pour cela l'achat (ou la récupération si on a un potager) de ces denrées est obligatoire. On se doit de ne pas trop acheter et de prévoir quelle sera notre consommation de la semaine si l'on fait les courses toutes les semaines.

Dans le cas des tests cela revient à prévoir son budget, préparer ses campagnes et les

Exemple de recette – Campagne

Recette Houmous	Campagne test « Houmous »
Ingrédients : <ul style="list-style-type: none"> • 500g de pois-chiches secs (féculents) • Eau (liquide) • Tahiné – crème de sésame (graines) • 2 citrons (légumes – fruits) • Sel, poivre (épices) • 2 gousses d'ail (condiment) • Huile d'olive (matière grasse) Outils : <ul style="list-style-type: none"> • Cocotte-minute • Robot mixeur • Presse citron • Pilon • Cuillères • Louche 	Tests à effectuer : <ul style="list-style-type: none"> • 500 Tests fonctionnels • 2 jours de Tests de fiabilité • ½ journée de Tests d'utilisabilité • 3 plateformes pour les Tests de portabilité • 15 critères sur les Tests de maintenabilité Outils : <ul style="list-style-type: none"> • Squash TM (ou un autre ALM comme XStudio ou Refertest) • Squash TA (ou un autre comme HP UFT) pour la portabilité • JUnit pour la maintenabilité • SonarQube et SonarLint pour la maintenabilité (ou autre outil comme Themis de ProMyze) • JMeter, (ou autre outil de mesure de performance comme Neoload ou Octoperf) pour la performance • 5 Environnements de test (pour tester en parallèle) • 1 Utilisateur pour les tests d'utilisabilité • Jira et le plugin XSquash TM (ou autre outil de suivi si besoin)
Faire tremper les pois chiches pendant 24 heures dans l'eau	Préparer les environnements de test pour les tests fonctionnels
Faire cuire à la cocotte-minute les pois-chiches et laisser siffler 35 minutes	Exécuter les tests fonctionnels automatiques
Piler les 2 gousses d'ail	Préparer l'environnement pour les tests de portabilité
Presser les 2 citrons	Préparer l'environnement pour les tests d'utilisabilité
Garder l'eau de cuisson des pois-chiches	Garder les logs des tests fonctionnels automatisés pour vérifier certaines performances
Récupérer les pois-chiches et les placer dans le robot	Exécuter les tests fonctionnels et analyser les résultats des tests automatisés
Ajouter l'ail, le jus de citron, du sel, du poivre et 3 louches d'eau de cuisson puis mixer le tout	Exécuter les tests d'utilisabilité et de portabilité
Lorsque l'ensemble fait une purée, goûter le résultat puis adapter (sel, poivre, eau, citron...) pour atteindre le goût souhaité	Lorsque les tests ont été exécutés et validés, faire des campagnes de tests exploratoires et les critères de qualité qui en ont besoin.
Mettre le houmous dans un saladier, faites la présentation puis ajouter l'huile d'olive	Faire des tests de maintenabilité et préparer le bilan
A noter : <i>Il est impératif de goûter pour assurer le bon goût du plat</i>	A noter : <i>Il est vivement recommandé d'ajouter des tests exploratoires et de vulnérabilité (par exemple avec Zap) dans les différentes campagnes de tests</i>

planifier. Cela est aussi important dans le test que dans la vie. En effet, une campagne non préparée (l'agilité n'échappe pas à la règle, y compris avec tests exploratoires qui nécessitent une charte) ne sera que peu efficiente !

Les restaurants : l'externalisation des tests

Avec les restaurants on externalise notre responsabilité de cuisiner. Il est possible de beaucoup manger, de manger équilibré, de manger selon un budget défini... Je dirais même plus qu'à force de sortir, on commence à avoir ses restaurants préférés.

Dans le cas d'une application, il est également possible d'externaliser ses tests.

L'externalisation des tests, même si elle est un vrai plus pour l'indépendance des tests, ne présage en rien de la qualité des tests qui seront fournis et de l'information qui sera récupérée. En fonction du budget alloué, il est possible, au restaurant, de manger plus ou moins en quantité mais aussi en qualité. C'est également le cas avec les tests. De même, à force d'externaliser ses tests des relations privilégiées seront nouées avec ses partenaires...

Je recommande néanmoins de ne pas tout externaliser. En effet, à part peut-être au Japon, il est plutôt rare de toujours manger au restaurant, et ce pour plusieurs raisons. On peut penser par exemple à la réactivité, au fait de toujours devoir se déplacer, au budget ou encore la connaissance à garder sur son produit.

La péremption des aliments : le paradoxe des pesticides

Les aliments sont des denrées périssables. Une tomate ou du lait frais ne sauraient être conservés plus de quelques semaines. A partir d'un moment, ces ingrédients ne peuvent plus être utilisés. C'est également le cas des tests. Les tests ont une durée de vie, faire toujours les mêmes rend ces tests inutiles, périmés !

Corps humain et application à tester

Depuis le début de l'article nous parlons de repas et de campagne de test. Or, comme vous le savez, il n'existe pas de campagne de test sans logiciel (ou application) à tester. De même il n'existe pas de repas sans personne pour le manger !

Dans cette analogie du logiciel à tester le « *System Under Test* » (ou « SUT » - le système à tester) est le corps du mangeur.

Comme nous le savons, un unique repas n'a généralement que peu de conséquences sur son corps. Il est possible de faire un excès ou même de sauter un repas sans que notre corps en soit réellement affecté.

De même, les besoins en apport énergétiques dépendent des personnes. Certaines vont devoir beaucoup consommer car leur corps dépense beaucoup, pour d'autres, c'est le contraire.

Ce besoin en apport énergétique dépend également de l'âge. Lorsque l'on est très jeune on mange moins que lorsque l'on est adolescent, puis on se met à avoir de moins en moins besoin de manger. Personnellement je mange 2 fois moins qu'à 20 ans et 4 fois moins qu'à 15 ans. Entretemps j'ai cessé de grandir et diminué fortement ma pratique sportive !

Avec les applications, de surcroît lorsque l'on a affaire à un logiciel construit de manière incrémentale, c'est pareil !

Au début de la vie de l'application il y a peu de test car peu de fonctionnalités à tester. Puis on atteint un point culminant (l'adolescence) avec un pic de développement et d'utilisation pour ensuite arriver à une lente diminution du besoin de test car l'application est mature et n'évolue plus beaucoup jusqu'à l'arrêt de cette dernière.

Dog fooding

Dans les années 70, une marque de pâtée pour chien vantait la qualité de ses produits en expliquant que leurs propres actionnaires en mangeaient !

Dans notre cas, c'est le cuisinier qui goûte ses propres recettes. Ce genre de pratique permet à l'entreprise conceptrice des applications de se rendre rapidement compte de la qualité de ses produits, pourvu que ses employés aient un réel besoin d'utilisation du produit à tester ; cela dit, on peut comprendre que le concepteur ait tout intérêt à se donner les moyens de tester lui-même les produits pour gagner du temps sur les délais de mise en production.

En effet, si les tests sont automatisés et fournis en même temps que le développement, on saura plus rapidement si le code est bon ou pas. Dans ce cas, le développeur est aussi le cuisinier, et il devient capable de

dérouler ses propres tests.

C'est avec cette mixité des rôles que les équipes agiles / DevOps fonctionnent. Traditionnellement, l'expression "on dit que l'on ne peut pas être juge et partie", ce qui a toujours justifié que le développeur ne pouvait pas tester ce qu'il avait codé, est maintenant challengée : si on sait à l'avance le goût et l'aspect et les étapes de la recette, rien n'empêche de cuisiner soi-même.

Cette pluralité des rôles pour un même profil est possible parce qu'on aura bien préparé les attentes autour du produit et la recette de test. C'est exactement ce qui se passe avec :

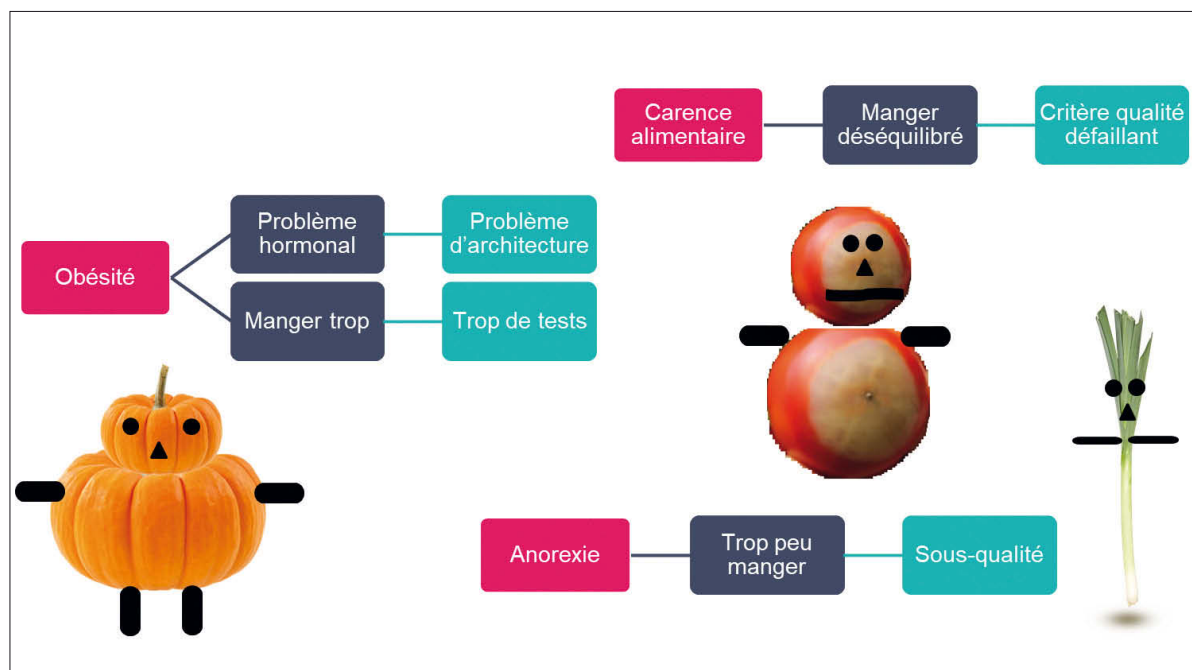
- les tests unitaires où le développeur réalise lui-même ses tests pour valider son code, par exemple avec l'approche TDD ou TCR ;
- les tests de type ATDD où le développeur réalise lui-même ses tests automatisés pour valider sa User Story à partir des scénarios fournis par le Product Owner.

Les maladies alimentaires et maladies des logiciels L'obésité : sur-qualité ou problème d'architecture

S'il est un mal bien connu lié à l'alimentation c'est celui de l'**obésité**. Il y a généralement 2 raisons liées à l'obésité.

La raison la plus instinctive est que l'on mange trop ce qui nous fait grossir. On ingurgite plus de calories que ce dont notre corps a besoin et notre corps stocke le surplus. Ce type de problème transposé aux tests revient à trop tester, c'est-à-dire à faire plus de tests que ce qui est nécessaire. En d'autres termes, faire de la **sur-qualité** ! Ce phénomène est courant notamment auprès de personnes qui veulent trop bien faire et essayent d'avoir des tests exhaustifs (ce qui n'est pas possible). Nous savons que l'obésité entraîne de graves problèmes de santé. C'est exactement le cas pour une application qui serait trop testée. Les tests coûtent de l'argent ! Le développement de l'application devient alors trop coûteux et cela peut entraîner jusqu'à la mort de l'application qui ne peut proposer suffisamment de revenus.

On peut également avoir des problèmes d'obésité à cause d'un dérèglement hormonal. Si l'on doit faire une comparaison avec les tests, ici le problème n'est pas la sur-qualité. Le problème est en effet plus pro-



fond et pourrait, par exemple, être lié à l'architecture du logiciel.

L'anorexie : sous-qualité

Sur le même principe il existe l'**anorexie**. Ce problème transposé aux tests revient à faire de la **sous-qualité** en testant trop peu et en laissant passer des anomalies majeures voire critiques, ce qui coûte généralement très vite très cher avec une sanction des utilisateurs.

Les carences alimentaires : défaut sur un critère de qualité

Il est toujours possible de souffrir de **carences alimentaires**, et ce, même si notre corpulence est tout à fait dans les normes. Cela s'explique par une alimentation pas assez équilibrée (ou correspondant aux besoins de notre corps), par exemple en ne faisant que des tests fonctionnels.

Dans les tests, cela revient à avoir une faiblesse importante sur un critère de qualité défini. On peut penser par exemple à un logiciel qui pourrait être très difficilement maintenable.

Ce problème peut s'avérer tout aussi dangereux pour une application que d'être victime d'une sur ou d'une sous-qualité. En effet si l'on prend l'exemple du critère de qualité de la performances, l'application peut être la meilleure du monde, mais si chaque page met 10 secondes ou plus à s'afficher, il y a fort à parier que cette appli-

cation ne rencontrera pas son public. On peut faire ce même parallèle avec l'ergonomie qui fait souvent la différence entre une application à succès et une application en échec.

Conclusion

Le testeur, qui se retrouve à devoir jongler entre différents tests afin de proposer les campagnes de tests les plus adaptées, se retrouve à faire un travail assez proche de celui du cuisinier. En effet, un cuisinier compose avec les ingrédients, les outils et le temps dont il dispose pour faire ses repas. Ces repas doivent de préférence être les plus adaptés possibles à ces différentes contraintes. Finalement, les utilisateurs du logiciel sont un peu les convives du cuisinier qui devra proposer des saveurs agréables et prendre soin à ne pas porter préjudice à leur santé.

Il est également bon de noter que le testeur, tout comme le cuisinier, ne peuvent assurer à eux seuls la qualité de l'application ou du repas. En effet, beaucoup de choses vont dépendre d'éléments externes comme le temps imparti, la connaissance des utilisateurs ou convives, les fournisseurs (avec le DevOps, les réactions des utilisateurs impactent directement les décisions d'implémentation des développeurs).

D'un point de vue plus général, la santé d'une application tout comme celle du corps humain se fait sur le long terme. La

santé d'une application découle de sa qualité et de son coût. La santé du corps humain dépend quant à elle de ce que l'on mange (ouf le prix n'entre pas vraiment en jeu !). Dans les 2 cas, une alimentation (des campagnes de tests) variée et équilibrée est nécessaire afin d'atteindre un équilibre vertueux (une bonne santé sur le long terme).

Pour finir, je souhaiterais ajouter un point qui me semble très important et qui n'a pas été abordé. Les tests doivent être inclus dès le début du cycle de vie de l'application, et continuer à s'adapter, évoluer avec la croissance de cette dernière. Il est inconcevable de commencer à nourrir un bébé 2 mois après sa naissance. Cela devrait être le cas avec tout logiciel ! De même l'alimentation des nouveaux nés est bien différente de celle des adultes. Les nouveaux nés mangent moins, plus souvent mais aussi moins varié. Cela correspond bien aux débuts des tests lors du développement d'une application : on commence avec peu de tests (peu de choses à tester), on les exécute souvent (nombreuses livraisons), ils sont peu variés (principalement fonctionnels)... Ce qui paraît logique avec notre analogie, la diversification alimentaire ne commence que plusieurs mois après la naissance.

Merci à Christophe MOUSTIER et Olivier DENOO pour leurs relectures et conseils qui ont fortement amélioré la qualité de cet article



TMMi : un modèle pour évaluer, faire progresser et certifier une organisation pour ses activités de tests

Le test logiciel est l'un des principaux facteurs de succès de tout projet informatique. Des compétences individuelles en test sont absolument nécessaires. Notamment celles reconnues par le vaste dispositif de certification ISTQB et ses presque 1 million de certifiés dans le monde, dont 30000 en France. Mais qu'en est-il pour évaluer la capacité d'une entreprise ou d'un centre de test à tester de façon rentable, efficace et contrôlée ? La réponse avec TMMi.

TMMi : une mine de bonnes pratiques spécifiques au test

Le modèle TMMi correspond à un ensemble de bonnes pratiques spécifiques au test et réparties dans des domaines de processus, eux-mêmes répartis sur des niveaux allant de 1 à 5. **1**

Comment évaluer votre niveau de maturité en test par rapport à un modèle reconnu ?

Les meilleures pratiques de l'industrie sur 5 niveaux



1 Les 5 niveaux de TMMi avec leurs domaines de processus.

TMMi, un modèle structuré et reproductible

Chaque domaine de processus est décliné en objectifs et pratiques spécifiques qui correspondent à des actions concrètes à réaliser pour être efficace dans l'activité de test associée. Le modèle TMMi est très complet et donne également des sous-pratiques et des exemples de livrables pour expliciter les bonnes pratiques attendues, comme dans l'exemple ci-dessous. **2**

La Fondation TMMi, un organisme à but non lucratif qui gère le dispositif complet

A cet ensemble de bonnes pratiques s'ajoute un dispositif complet d'évaluation et de certification règlementé par la fondation TMMi. Le site www.tmmi.org fournit tous les documents et toutes les informations nécessaires à une bonne compréhension du modèle TMMi et de son utilisation, et notamment :

- Le modèle TMMi lui-même, avec la description détaillée de toutes les bonnes pratiques ;
- La liste des organisations déjà certifiées TMMi, comme en France BNP Paribas Cardif et Crédit Agricole Payment Services ;
- La liste des sociétés habilitées à réaliser des audits TMMi officiels, comme en France la société Certilog ;
- La liste des personnes ayant les statuts d'évaluateurs TMMi, « Assessor » ou « Lead Assessor ».

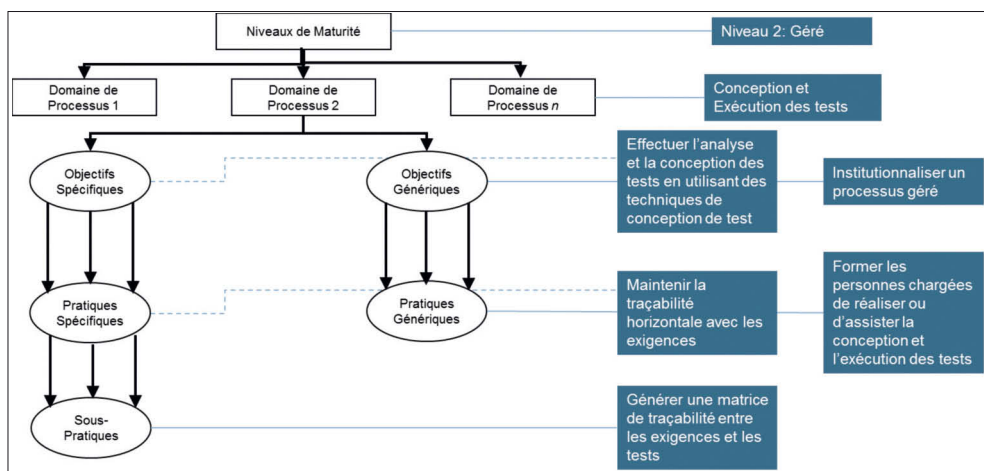
Trois phases principales pour un audit TMMi

L'audit d'un centre de test se déroule généralement en 3 phases présentées dans le schéma ci-dessous. Il est important de noter que la mise en œuvre du plan d'amélioration produit par la phase d'évaluation est principalement portée par les membres du centre de test évalué, avec un soutien ponctuel de l'évaluateur TMMi. **3**

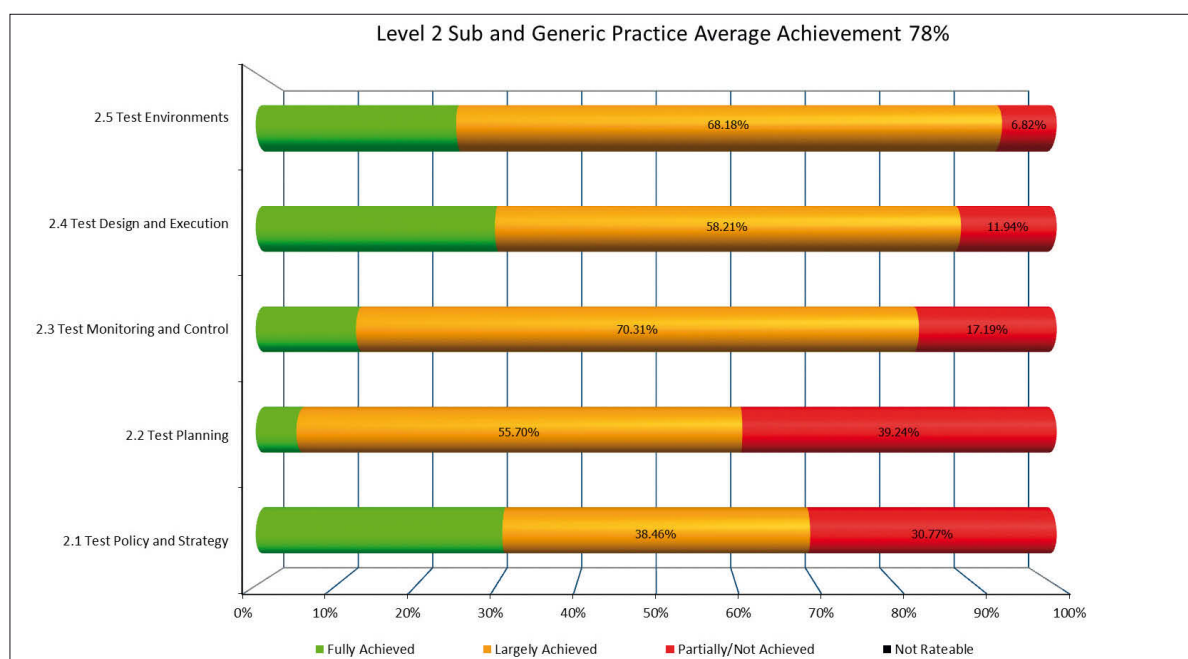
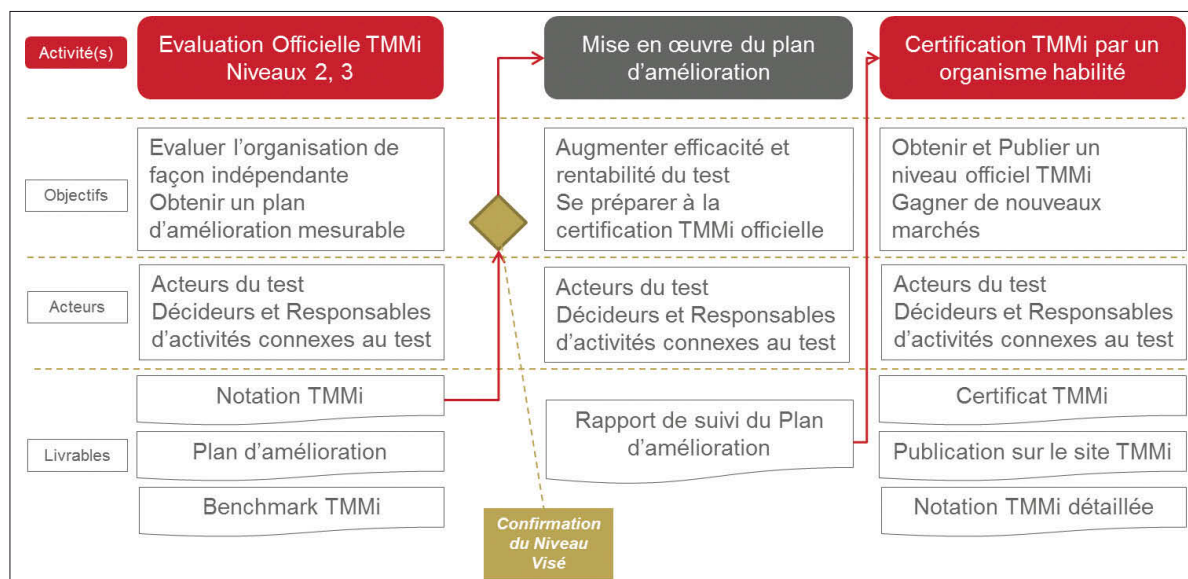
Des livrables concrets à la fin de l'audit

Les principaux livrables d'un audit sont :

- Une notation globale TMMi qui précise le pourcentage de mise en œuvre efficace des bonnes pratiques pour chacun des domaines de processus du ou des niveaux évalués (voir l'exemple ci-dessous) ;
- Un plan d'amélioration qui précise les actions à mettre en œuvre pour atteindre un certain niveau de maturité, avec, pour chaque action, son coût de mise en œuvre et l'estimation de sa valeur ajoutée pour le centre de test ;
- Le certificat TMMi, dans le cas d'un audit formel allant jusqu'à l'obtention de la certification. **4**



2 Les différents composants du modèle TMMi



TMMi, un modèle qui évolue

Le modèle TMMi, même s'il a toujours été applicable à tout type d'organisation et à tout type de mode de développement, séquentiel ou agile, est régulièrement mis à jour et enrichi de nouvelles bonnes pratiques. Par exemple, une déclinaison du modèle vient d'être produite pour les organisations fonctionnant en mode Agile.

La communauté TMMi en France et les « certifiés TMMi »

Le modèle TMMi a près de 15 ans et depuis 2016, nous observons une accélération de

son utilisation dans le monde et aussi en France.

Il existe une formation certifiante TMMi dont l'objectif est de bien expliquer le mode de fonctionnement du modèle. En effet, celui-ci est à la fois simple et sophistiqué. Il est important de comprendre précisément son mécanisme pour être capable de s'auto-évaluer, en tant que responsable de test, ou d'évaluer le centre de test d'un client, en tant qu'auditeur. A ce jour, plus de 100 personnes en France sont déjà des professionnels certifiés TMMi et utilisent régulièrement le modèle.

Par ailleurs, afin de faciliter l'usage de TMMi dans le monde, la fondation TMMi a

mis en place un ensemble de communautés locales chargées de représenter et développer TMMi dans les différents pays du monde. En France, c'est le CFTL (Comité Français des Tests Logiciels) qui porte cette responsabilité. Un groupe est animé par 3 experts en test : Marc SOUNE-SEYNE, Françoise BARBEAU et moi-même. Une première journée TMMi est déjà planifiée à Paris le 26 mai 2020 avec un retour d'expérience du responsable du centre de test de BNP Paribas Cardif et différents ateliers de travail. Tous les certifiés TMMi sont invités ! Pour tout renseignement sur TMMi, vous pouvez vous adresser directement au CFTL, en utilisant l'adresse info@cftl.fr

Accordons-nous autant d'attention à la spécification des tests qu'à la spécification des exigences ?

Serge Dubois, Technical Manager - Polarsoft

Les tests sont les principaux outils employés pour vérifier et valider les exigences. On peut cependant trouver un double sens à ce verbe « Valider » (en référence à l'éternelle question « quelle est la différence entre vérifier et valider ? ») : le premier, s'assurer que l'exigence répond aux besoins du client, le second, s'assurer de la compréhension et l'acceptation de l'exigence par les différentes parties prenantes. Pour collaborer sur cette compréhension, la relecture, la revue des spécifications d'exigences par les pairs est une pratique courante, voire systématique, l'étape de revue est obligatoire dans le cycle de vie de l'exigence. Avons-nous cette même approche pour les tests et documents associés ?

La dernière exécution des tests, combien de « Passed », combien de « Failed » ?

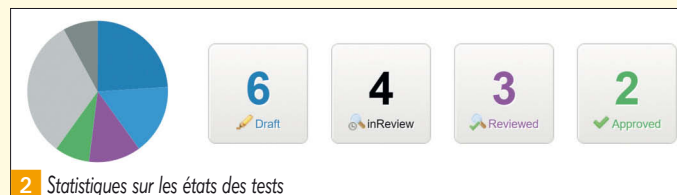
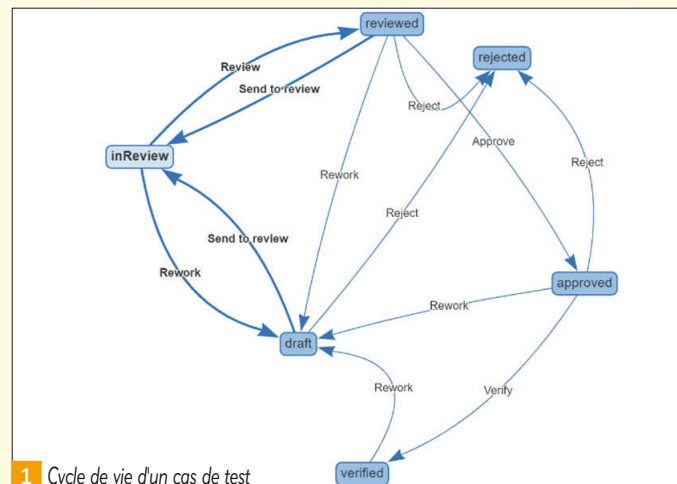
Si je surprénais une discussion « à la machine à café » entre des développeurs et des testeurs sur les tests logiciels en cours, j'aurais tendance à penser que les thèmes principaux abordés porteraient sur l'exécution des tests, leurs résultats, les anomalies identifiées, plus que sur la définition, la rédaction, la pertinence ou la « justesse » de ces tests.

Vous me répondez que cela dépend de nombreux facteurs, comme le secteur d'activité (on peut penser que les plans de validation sont fortement documentés pour des systèmes critiques comme les dispositifs médicaux, l'aéronautique, etc.), l'avancement du projet (ex. : on ne parle pas encore d'exécution, car les développements n'ont pas commencé), la conduite du projet (cycle en V, agile ou hybride), la place des acteurs du testing dans l'équipe projet (implication au plus tôt ou tardive), et vous auriez raison !

Vous ajouteriez que nous nous attachons à d'autres aspects comme la couverture des exigences par les tests, mesure que nous apporte la traçabilité entre ces artefacts. Mais qu'en est-il de la lisibilité, la compréhension, la pertinence du test lui-même ?

Donner plus d'importance à l'écriture des tests

Je pense que comme je l'écrivais dans un précédent article dans Programmez!, « qu'il est possible d'appliquer aux spécifications des tests les mêmes bonnes pratiques que pour les spécifications des exigences, à savoir élucidation,



documentation, revue par les pairs, approbation... ».

Nous pourrions penser que cela va alourdir la charge du projet (les tests sont encore parfois une « variable d'ajustement » de la charge et coût global du projet), je pense qu'au contraire mieux travailler en amont sur la qualité des tests va permettre de réduire les aller-retour tardifs entre testeurs, développeurs et analystes métiers. Il y a aura bien entendu toujours des échanges sur les résultats des tests, les dysfonctionnements rencontrés, mais moins sur la pertinence, la justification et la justesse du test.

Quels moyens mettre en œuvre pour répondre à cet objectif Un workflow adapté. 1

Les utilisateurs de Polarion ALM

mettant en œuvre la solution pour spécifier les exigences du projet, couvre naturellement le processus de relecture et de revue des spécifications d'exigences en employant (ou personnalisant) tout d'abord un Workflow ad hoc, que ce soit au niveau des Work Items Exigences, ou au niveau des documents LiveDocs dans lesquels sont décrites ces exigences. Ceux qui utilisent la plate-forme Polarion pour spécifier les tests disposent et utilisent les mêmes outils pour cas de tests et documents liés aux activités de tests (plans, protocoles, cahiers de tests, etc.), à savoir des Work Items « Cas de tests », des LiveDocs, et des Workflows attachés. Il n'y a pas de discontinuité numérique ni de freins liés à la connaissance de l'outil, car

les mêmes moyens sont employés. On peut voir ce Workflow comme la porte d'entrée de la revue des tests par les pairs ou parties prenantes : « Quels sont les cas de tests en cours d'écriture, ceux qui sont en attente de revue, et ceux prêts pour l'exécution ? »

À noter que dire qu'un test est dans un état Passed ou Failed est un raccourci assez courant, c'est en réalité l'exécution du test, dans le cadre d'une campagne pour un environnement et un périmètre donné qui nous donne cet état (on parle dans Polarion ALM de TestRecords). Nous pouvons donc définir les états d'un test, incluant ces étapes de revue, mais nous ne voyons pas à ce niveau les états « passed et failed » (ces états seront présents, mais pour leur exécution dans un Test Run). 2

La relecture et revue des documents des tests et cas de tests 3

Considérons la relecture et la revue proprement dite. La collaboration entre les pairs (les testeurs) ou avec les autres parties prenantes sous Polarion ALM peut se matérialiser sous différentes formes, mais considérons d'abord simplement les échanges par les commentaires :

- Un testeur ou responsable des tests peut commenter les cas de tests et/ou documents de tests en cours de rédaction par ses pairs : toutes les données propres à ces revues, tout l'historique des conversations est enregistré dans le référentiel (Audit Trail complet), certes des notifications de revues peuvent être adressées par mail,

mais l'essence de la revue (les commentaires), et les éventuelles réécritures et améliorations des tests qui en découlent, sont sauvegardées de manière immuable dans le référentiel de Polarion.

- Les analystes métier et développeurs peuvent également disposer des droits pour relire et commenter ces tests, leurs connaissances approfondies des besoins exprimés par le client d'une part, et l'implémentation réalisée d'autre part sont précieuses pour élaborer des tests pertinents. Pour appuyer cette revue, les testeurs ont eux aussi un accès direct en lecture, et en un clic aux spécifications d'exigences (car présentes sous forme de LiveDocs Polarion), et pas uniquement aux exigences systèmes ou logicielles, mais aussi aux exigences clients leur permettant ainsi d'avoir une vision plus globale de ce qu'ils doivent tester.

Ouvrir la spécification aux commentaires est une première approche pour collaborer sur les assets de tests et améliorer leur qualité, notons qu'il est possible d'être plus formel dans cette revue. Le choix peut porter sur la mise en place d'un cycle de vie du test conditionné par des approbations. Dit autrement, il ne s'agit plus « d'inviter » à la revue, mais d'intégrer formellement cette étape dans le processus de définition des tests.

Une documentation exhaustive dans les environnements réglementés

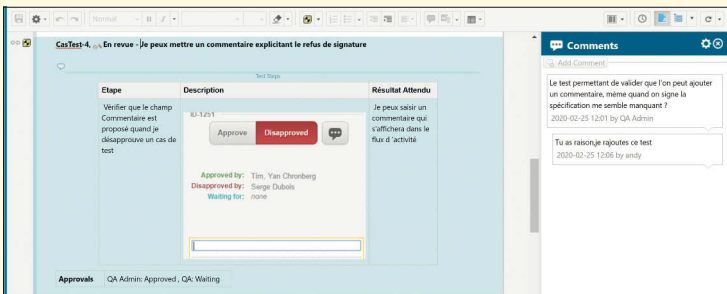
Dans certains domaines, l'importance (ou l'obligation !) de cette revue des tests ne fait pas débat. La vérification et la validation des systèmes et logiciels critiques dans les environnements réglementés, comme l'aéronautique (DO 178C), les dispositifs médicaux (FDA validation guidance) ou encore pharmaceutiques (bonnes pratiques GAMP 5) pour ne citer qu'eux, se doivent d'être richement documentées. Prenons l'exemple de GAMP, le modèle documentaire proposé dans le guide GAMP® Good Practice Guide : Testing of GxP

Systems comprend les plans de test (vision stratégique), des spécifications ou protocoles de tests pour chaque phase, les cas tests avec données entrantes et résultats attendus, et pour la vision exécution, de nombreux rapports synthétiques ou détaillés.

Au-delà de cette documentation, la revue des tests est clairement mise en avant : le GAMP invite à **revoir et approuver** les spécifications des tests avant le début de leur exécution - avec bien entendu des relecteurs qui ne sont pas les auteurs des tests !). Ceci requiert de la collaboration, un processus bien établi avec responsabilisation, des moyens de spécifications mettant l'accent sur les données et leur qualité. L'ensemble de ces éléments sont fédérés dans le référentiel de Polarion ALM. De plus, ces entreprises étant régulièrement contrôlées par des organismes de certification, ils disposent de pistes d'audit complets pour démontrer ou prouver la qualité de leur processus de vérification et validation, puisque tous changements, toutes revues ou approbations sont historisées automatiquement et immédiatement accessibles.

Les responsables de tests peuvent s'engager par la signature électronique

Autre obligation fréquente dans ces environnements réglementés, la mise en conformité avec la norme 21 CFR Part 11 qui s'attache à garantir l'authenticité, la fiabilité et la confidentialité des enregistrements électroniques et de leurs signatures. Puisque sous Polarion ALM les documents sont numérisés (LiveDocs), les signatures le sont tout autant. Polarion permet de capturer ainsi l'engagement des parties prenantes que ce soit sur les spécifications ou encore sur les rapports d'exécution des tests. Nous ne sommes pas sur la relecture et la revue à proprement parler, mais faire signer électroniquement les spécifications de tests (et avoir à disposition en un clic les différentes versions signées) nous permet de contractualiser notre compréhension des besoins de tests en concertation avec ceux et celles



3 Revue des spécifications de tests dans un LiveDoc



4 Revue par les approbations des tests



5 Signature électronique des documents et Audit Trail par version

qui nous ont fournis le Quoi et le Comment de l'implémentation.

Un clair distinguo entre spécification des tests et exécution des tests

Cette approche s'inscrit dans l'approche assumée de Polarion ALM de bien séparer, pour mieux les considérer, spécification des tests, et exécution de ces tests. Avoir une plate-forme unifiée, cohérente en termes de moyens à disposition quelle que soit le profil d'acteur, facilite la collaboration et la rend plus efficiente. Certes, le test n'est plus (ou tend à ne plus l'être !) le parent pauvre du développement

logiciel, en agilité notamment les testeurs sont des acteurs majeurs de l'équipe agile, il n'empêche que la documentation des tests se limite parfois à une liste de cas de tests tracés vers des exigences, et surtout la revue des tests nous semble moins prônée que la revue des exigences, qui elle, ne fait pas débat. Cela n'est pas une question d'outils à disposition, plus une question de pratiques, disposer de solutions homogènes permettant de mieux conduire, structurer et piloter ces activités de revues peut cependant ouvrir la réflexion sur ce sujet. Pour plus d'information sur Polarion ALM : www.polarsoft.fr

**Marc Hage Chahine**

Key member centre d'expertise de test d'Altran (ITQ)
Créateur et animateur du blog La taverne du testeur

**Benjamin Butel,**

coach test chez Klaxoon.
Auteur d'articles dans la Taverne du testeur.
Passionné de test logiciel
Certifié ISTQB Full advanced et automatisation.

Entretien avec Daisy Draté ingénieure qualité sur l'application AVoKa

Dans cet article, Marc Hage Chahine et Benjamin Butel nous proposent de faire l'éloge des différents contributeurs à un logiciel de qualité au travers d'une interview d'un produit fictif.

Depuis quelques années, le manger local, sain, avec des produits frais refait surface, et les gens se remettent de plus en plus au jardinage. Pour autant, les dernières générations sont moins à l'aise avec la gestion d'un potager. J'ai la chance d'avoir avec moi, Daisy Draté, qui a participé à la construction de l'application « AVoKa » qui fait fureur !!! Cette application propose des conseils personnalisés pour optimiser ses plantations, que cela soit dans un jardin ou en pots : choix des semences, gestion de l'arrosage, choix des engrais naturels, dates de plantation, entretien de la terre ...

Daisy, qui êtes-vous et quel est votre rôle dans la construction de cette application ?

Je m'appelle Daisy et je suis une passionnée de test logiciel.

Mon rôle de testeuse sur AVoKa a été d'apporter ma vision sur le produit, de travailler sur les différents cas d'usage pertinents, et moins pertinents, mais surtout d'apporter à l'équipe la philosophie de la qualité. Attention, la qualité ce n'est pas forcément toujours faire plus, il faut savoir jusqu'où aller. Pour cela, on peut dire que j'ai exercé un rôle de Coach.

Mon rôle de spécialiste test était principalement de concevoir et exécuter les tests, mais cela ne s'arrêtait pas là ! En tant que membre de l'équipe, j'ai aussi contribué à différentes tâches qui, au premier abord, ne sont pas liées à mon métier. J'ai par exemple fait de nombreuses revues de code, participé à l'écriture des spécifications, et même participé à la mise en place des outils de mesures.

Comment l'idée de ce produit a-t-elle germé ?

Le créateur de la société, Judas Sperge, passe beaucoup de temps dans son jardin. Malheureusement, il n'a pas vraiment la main verte, et trop souvent, il n'arrivait pas à trouver des conseils avisés qui lui convenaient. Il a constaté que son entourage faisait le même constat : trop d'informations sur Internet, trop généralistes ou trop spécifiques, de même pour les livres spécialisés.

Il s'est entouré d'un chargé d'étude marketing pour étudier le terrain, si je peux me permettre l'expression.

Il se passe du temps entre l'idée et le produit fini. Comment avez-vous fait pour réaliser la première version en six mois il me semble ?

Oui, six mois c'est court, mais l'étude a fait ressortir les besoins utilisateurs les plus attendus.

Avec Judas, le responsable produit (qu'on appelle PO pour Product Owner chez nous), et l'équipe de réalisation, nous avons concentré nos efforts pour répondre à ces besoins en priorité tout en proposant une interface instinctive, et des temps de réponses très rapides. Dans l'équipe, nous détectons devoir attendre pour avoir accès à l'information

Comment gérez-vous les différentes demandes des utilisateurs ? Certaines demandes sont sûrement contradictoires néanmoins à chaque nouvelle version, tout le monde y trouve son compte.

C'est très simple. Notre PO a travaillé avec notre analyste métier, Thibault Range, pour comprendre les différents besoins et de-

mandes, afin de les regrouper et de bien les concevoir. De même, les données d'utilisations recueillies permettent de détecter certaines difficultés rencontrées par les utilisateurs.

Ensuite, la totalité de l'équipe donne son avis sur ce besoin et contribue à la priorisation.

On imagine les usages derrière. On essaie de se mettre à la place des utilisateurs. Chacun dans l'équipe se met dans la peau d'un profil utilisateur, persona, et évalue ce besoin. Est-il pertinent ? Quand et dans quel contexte ? Comment je le mets en œuvre ?

Tout cela aide à formaliser les fonctionnalités que l'on va développer pour répondre au besoin.

Mon voisin et moi utilisons tous les deux l'application. Les conseils sont différents mais nous correspondent bien. Comment avez-vous réussi une IA si pertinente ?

Notre analyste métier nous apporte les éléments clefs qui définissent le jardinage. Puis notre expert en données (notre data scientist) a analysé tout ça pour en déduire la meilleure gestion de données possible.

Le data scientist a ensuite testé différents algorithmes et entraîné différents réseaux de neurones afin de proposer des résultats cohérents. Il me semble également important de noter que cette IA continue d'évoluer afin de s'améliorer.

La pertinence des résultats proposés est le fruit d'une maîtrise des données métier transformées en données fonctionnelles puis en données techniques (coté persistance).

L'application est super intuitive, comment avez-vous fait ?

Nous avons une championne du design Marie Aubrosse (l'UX designer) qui a proposé des interfaces qui ont été bien implémentées par l'équipe, et particulièrement nos intégrateurs.

Grâce à eux, l'appli propose à la fois une expérience fluide, intuitive et esthétique.

Un des besoins était que l'appli soit accessible aux personnes en situation de handicap, dans son jardin et agréable à utiliser. Il faut avoir envie d'y revenir ! Il fallait donc que l'application puisse s'adapter aux contraintes d'un usage en extérieur et pour le plus grand nombre de personnes.

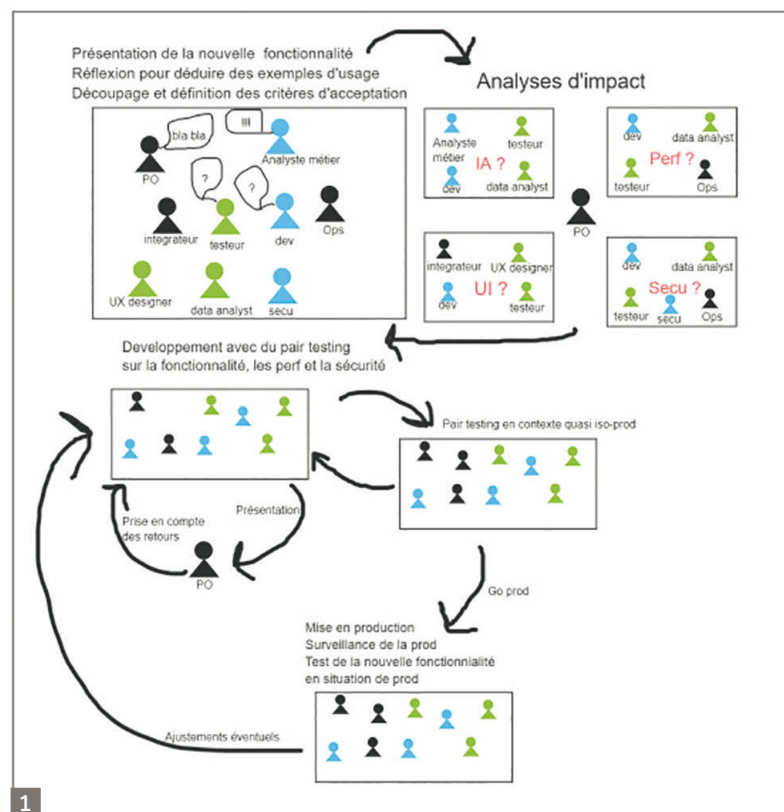
J'utilise l'application tous les jours mais je ne tombe sur aucun bug! Tout fonctionne super bien, quelle est votre recette ?

(rire) Je suis une superbe testeuse !!!

En vérité, c'est un travail collaboratif. Le fait de définir et concevoir en équipe le produit contribue énormément à ce résultat. Les développeurs sont experts dans leur domaine et participent à l'effort de test. Les tests unitaires et d'intégration sont pertinents, ils font aussi pair testing avec moi. Enfin, toute l'équipe fait des sessions de tests exploratoires. Au-delà de la recherche du bug, cela nous permet d'évaluer ce que nous avons imaginé pour satisfaire le besoin. Parfois, nous ne sommes pas convaincus, on jette tout simplement et on refait différemment.

L'application est évolutive et collaborative, comment implémenter une telle fonctionnalité ?

Cette fonctionnalité est probablement celle qui nous a demandé le plus de travail, mais elle nous semblait essentielle ! La mise en place des méthodes Agile est là pour pouvoir répondre à des changements comme des changements de contexte. Le contexte évolue très vite, de nouvelles plantes apparaissent, le climat évolue, notre application se devait d'évoluer en temps réel. Pour cela, après discussions dans l'équipe, nous avons décidé d'utiliser l'intelligence commune de nos utilisateurs. C'est à partir de là que Marie, notre UX designer, et Thibault, notre analyste métier, ont collaboré pour proposer l'interface ac-



tuelle avec les fonctionnalités que vous connaissez.

Les développeurs front, back et le data scientist, qu'on surnomme « Doc », ont ensuite fait ce qu'il fallait pour que l'idée de départ soit bien implémentée.

Rien ne vaut un schéma pour illustrer une idée. Voici à quoi ressemble la mise en place d'une fonctionnalité chez nous : **1**

Selon les statistiques des derniers téléchargements, l'application est déjà utilisée par plusieurs centaines de milliers d'utilisateurs dans plus de 20 pays...en à peine 6 mois. Comment gérez-vous cette croissance ?

Là encore, c'est un travail d'équipe. Tout d'abord, l'internationalisation a été pensée dès le début dans la conception du produit, afin de séparer les traductions du cœur du métier de l'application.

Cela nous permet de rester flexible et de s'adapter aux demandes selon la provenance des clients.

Nous avons aussi les OPS qui ont donné leur vision pour satisfaire ce besoin d'internationalisation.

Avec les capacités d'hébergement d'aujourd'hui, il est possible de rendre évolutive

la croissance de l'usage mais aussi de la régionaliser. Tout ceci a été pris en considération dès la conception de l'architecture.

Enfin, afin d'assurer la qualité de notre application en dehors de la France, nous avons eu recours au Crowdfunding. Les Crowdfunders ont pu tester l'application dans plus de 10 pays ciblés, et, en plus de nous donner leur ressenti en tant que « bêta testeurs » : ils avaient pour mission de nous faire des retours sur la qualité des traductions proposées.

Comment avez-vous adressé la performance ?

Comme dit précédemment, les OPS ont participé à la conception de l'architecture. Cela a permis de mettre en place une architecture robuste, modulable et facilement maintenable avec le besoin croissant d'utilisateur. Cela assure aussi une haute disponibilité de l'application.

Doc nous a également permis d'optimiser les usages des bases de données pour servir une rapidité de traitement des demandes utilisateurs.

Enfin, dès le début, nous avons identifié les niveaux de performance que nous souhai-

tions atteindre. Cela a permis de mettre le bon niveau de test à chaque niveau du développement : analyse statique du code pour chasser des mauvaises pratiques entraînant des complexités trop importantes, des tests unitaires de performance, tests de performance sur les API, sur le front, et, enfin, sur le système complet. Chaque catégorie de test cherche à identifier des problèmes différents de performance et c'est le cumul de l'ensemble qui fait qu'on satisfait une haute qualité sur cet aspect.

C'est une des premières fois où je n'ai aucune lenteur sur une application même avec mon téléphone super vieux, c'est impressionnant...

Oui, du côté de l'application, nos experts en développement sur Android et iPhone ont aussi livré leurs contraintes en début de projet pour que les performances applications ne soient pas un « pain point » (N.D.L.R : point de frustration). De même, nous avons également fait très attention à la consommation d'énergie afin de sauvegarder au maximum la batterie des smartphones utilisant notre application. Il aurait été dommage d'avoir une qualité de haut niveau sur tout le reste et se louper sur la partie la plus proche de l'utilisateur. Le travail entre l'UX, les designers et l'ensemble des développeurs a permis de trouver le meilleur compromis pour garder une expérience utilisateur de grande qualité.

Pour proposer des conseils aussi avisés, je suppose que vous avez des partenaires. Comment gérez-vous la récupération de leurs données ? Et comment gérer leur évolution ?

En effet, nous avons quelques partenaires qui proposent des API pour récupérer de l'information en temps réel. Nos développeurs ont travaillé pour intégrer proprement les API et notre data analyst a fait un gros travail pour proposer un modèle permettant d'intégrer ses données, évaluer leur fiabilité afin que l'IA puisse réagir en conséquence. Enfin, nos OPS sont capables de détecter lorsqu'un partenaire est en défaillance ou a fait évoluer le contrat d'interface. Tout a été pensé pour gérer ces cas de figures et éviter que l'IA s'en trouve complètement perdue.

Et la sécurité ? On n'en parle pas beaucoup mais vous gérez tout de même de nombreuses données.

Notre expert en sécurité et notre data analyst ont travaillé ensemble afin de séparer les données dites publiques des données sensibles (comme les données utilisateur). Cela a permis d'intégrer cela dès la conception de l'architecture logicielle. Notre expert sécurité a aussi accompagné les développeurs sur les bonnes pratiques des usages du WEB et fait des pen tests réguliers pour identifier des problèmes de sécurité potentiels.

Une analyse des dépendances logicielles a

aussi été faite pour réduire le risque potentiellement lié à ces dernières (risque sécurité mais aussi de licence).

Enfin, notre expert sécurité a travaillé avec les OPS afin de mettre en place un système de monitoring pour détecter des tentatives d'intrusion ou de DDOS afin de réagir rapidement.

En conclusion, on peut dire que le succès de votre application est collectif
Oui complètement.

La qualité est l'affaire de tous, on ne peut pas proposer un logiciel de qualité sans l'apport de l'ensemble des acteurs qui contribuent à le construire. Regardez, si on prend les critères qualité de l'ISO-25010, un profil seul ne peut pas tout faire : **2**

Et le mieux dans tout ça, c'est qu'on a pris du plaisir (et on continue à en prendre) à travailler ensemble. Tout n'a pas été simple. Il y a eu des moments difficiles, mais à chaque fois, le collectif était là pour tirer tout le monde vers le haut. Et honnêtement, je pense que ça se voit dans le résultat.

Désormais, place à la suite mais chut, c'est confidentiel (rire) !!!

Merci Daisy de nous avoir éclairés sur la genèse de votre application. Je retiens que la clef de la réussite passe par l'anticipation, la pluridisciplinarité des expertises, le collectif, et aussi le plaisir et le fun.

Quel rôle sur quel critère qualité ?

2

Fonctionnel	Performances	Compatibilité	Utilisabilité
<ul style="list-style-type: none"> Analyste métier Product Owner (PO) Commercial Développeurs front et back Testeur Data scientist et analyst 	<ul style="list-style-type: none"> Ops Développeurs back Analyste métier (specs non fonctionnelles) Testeur 	<ul style="list-style-type: none"> Analyste métier Développeurs front et back Crowd testeurs Ops 	<ul style="list-style-type: none"> Intégrateur Ergonome (UX designer) Développeur front Crowd testeurs PO Testeur
Fiabilité	Sécurité	Maintenabilité	Portabilité
<ul style="list-style-type: none"> Testeur Analyste métier Développeurs front et back Ops Data scientist Intégrateur 	<ul style="list-style-type: none"> Développeurs front et back Ops Expert sécurité 	<ul style="list-style-type: none"> Testeur Développeurs front et back Ops 	<ul style="list-style-type: none"> Analyste métier Testeur Crowd testeur PO Développeur Iphone Développeur Android Commercial



Anthony Giretti
MVP, MCS D
Developer, Blogger, Speaker
anthony.giretti@gmail.com
<http://anthonygiretti.com>



Intégrez NDepend à vos projets .Net Core !

En informatique et en génie logiciel en particulier, la qualité des logiciels est une évaluation globale des logiciels, basée sur de nombreux indicateurs. L'exhaustivité des fonctionnalités, la correction et la précision des résultats, la fiabilité, la tolérance aux pannes, la facilité et la flexibilité de son utilisation, la simplicité, l'extensibilité, la compatibilité et la portabilité, la facilité de correction et de transformation, la performance, la cohérence et l'intégrité des informations qu'il contient sont autant de facteurs de qualité. En génie logiciel, la factorisation des données et du code est le moyen universel d'obtenir la qualité. Nous parlerons dans cet article de NDepend, un outil pour aider les développeurs de logiciels à atteindre un haut niveau de qualité de leurs programmes.

Qu'est-ce que NDepend ? ¹

NDepend a été créé par Patrick Smacchia en 2004. Cet outil a été commercialisé en 2007 et a beaucoup évolué depuis. Il propose une analyse de code pour vos applications .NET et .NET Core et vous pouvez l'essayer gratuitement pendant 14 jours. Il peut être utilisé en mode Visual Studio autonome ou intégré. NDepend vous donne un accès en un clic à de nombreuses métriques dans votre code avec une représentation graphique. Intéressant ! Avec le tableau de bord, la fonctionnalité la plus intéressante de NDepend est le langage de requête sur lequel il est construit : CQLINQ. Mais avant de jeter un œil, attachons-nous sur le processus d'installation.

Installation

L'installation de l'extension NDepend pour Visual Studio est vraiment facile, téléchargez-la à cette adresse <https://www.ndepend.com/download> et installez-la comme ceci en cliquant sur Visual Studio Extension Installer, puis sélectionnez la version de Visual Studio sur laquelle vous installez : ² L'excellente chose à propos de NDepend est la facilité avec laquelle vous pouvez commencer à l'utiliser et l'excellente documentation disponible sur leur site Web. Il y a

des vidéos en ligne, des tutoriels, des captures d'écran, des explications étape par étape et vous ne pouvez pas vous tromper. Pour plus de détails, vous pouvez vous rendre à cette adresse : <https://www.ndepend.com/docs/getting-started-with-ndepend#Tuto>

Exécuter NDepend

Vous pouvez exécuter NDepend en mode autonome avec l'exécutable VisualNDepend.exe (dans le même répertoire d'installation que l'exécutable de l'extension Visual Studio) ou exécuter l'extension dans Visual Studio à partir du menu : ³

1- Le dashboard ⁴

Sur le même écran du dashboard, NDepend fournit de nombreuses métriques comme :

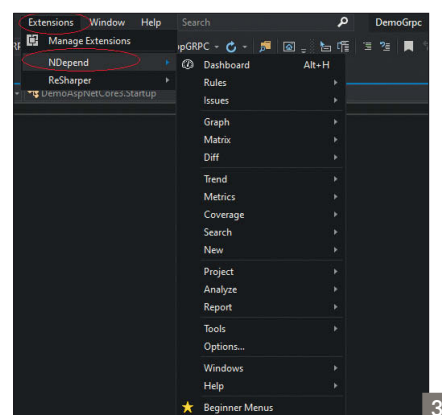
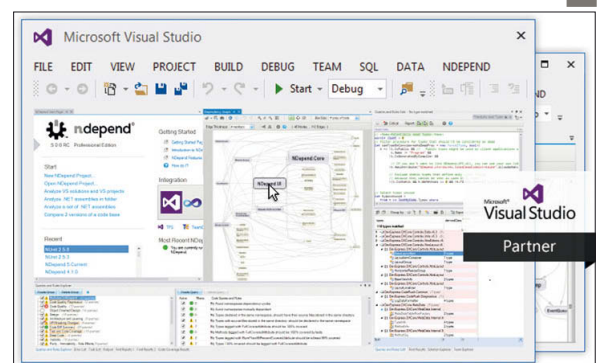
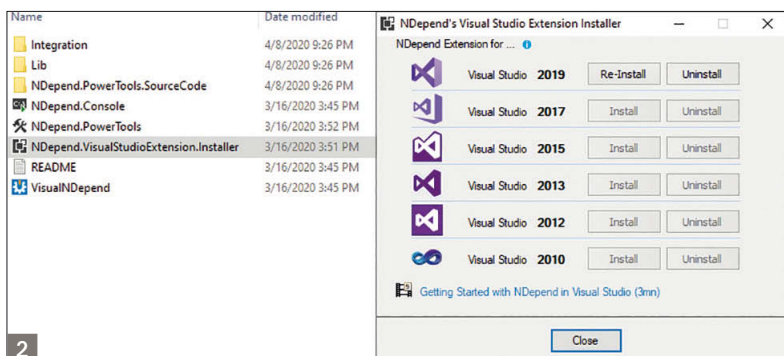
- La dette technique ;
- Les lignes de code ;
- Les commentaires ;
- Les différents niveaux de problèmes dans le code ;
- Les règles de programmation ;
- La couverture de code ;
- Les quality gates.

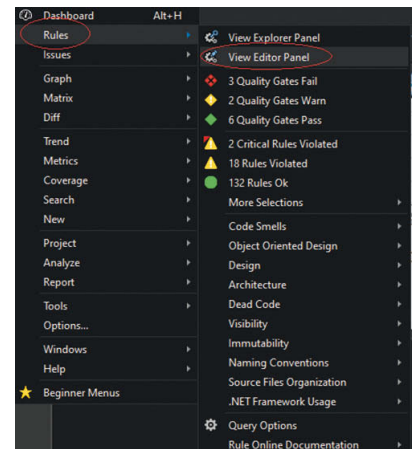
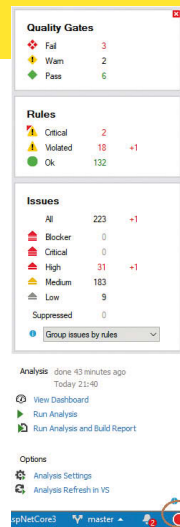
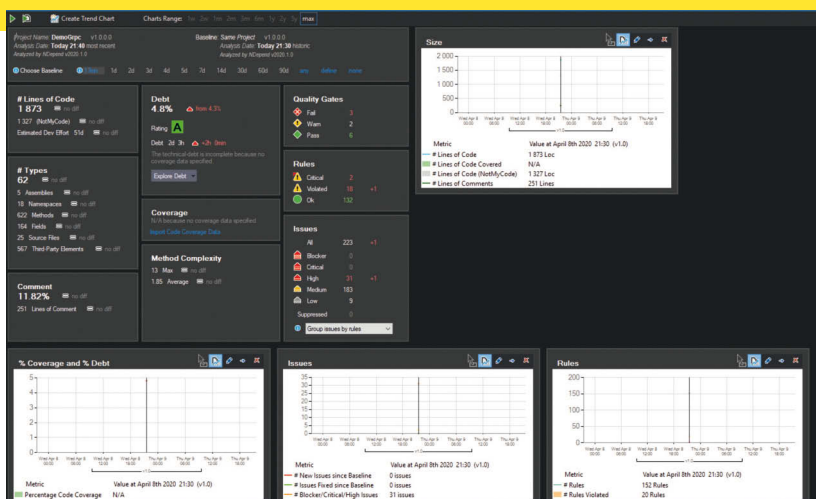
Et bien d'autres fonctionnalités ! Chaque métrique est cliquable pour approfondir. Il fournit une analyse vraiment énorme de

votre code. Vous pouvez à tout moment jeter un œil à un petit résumé et recalculer votre tableau de bord en cliquant sur le cercle coloré en bas à droite de votre écran, très pratique ! ⁵

2- CQLinq

CQLINQ, qui signifie Query Code LINQ, permet d'écrire des requêtes C# LINQ sur un modèle abstrait de votre code. Concrètement, vous pouvez interroger votre code de la même manière que vous interrogeriez les vues ou les tables d'une base de données. Par exemple, pour rechercher tous les constructeurs de classes, j'ai écrit la requête suivante :

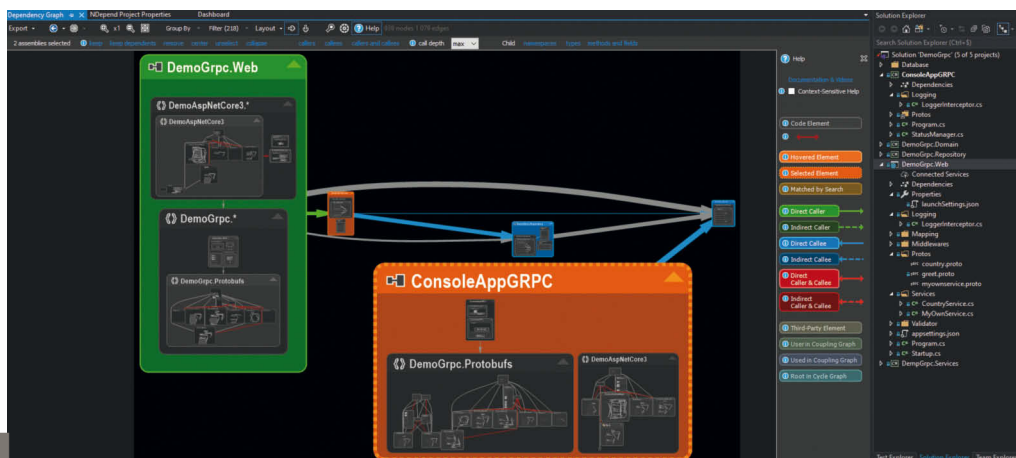




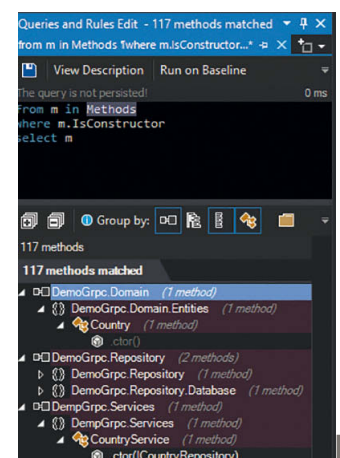
4

5

6



9



7

from m in Methods
where m.IsConstructor
select m

Si vous souhaitez écrire votre requête, allez dans le menu NDepend, sélectionnez «Rules» et «View Editor Panel»: 6
Il ouvre une petite fenêtre où vous pouvez entrer votre requête. Dans le même temps, le résultat est affiché en bas. Pas mal non ? 7

3- Le graphique de dépendance

Grâce à NDepend explorer l'architecture de code existante dans Visual Studio via le graphique des dépendances se fait en quelques clics, allez dans le menu NDepend, sélectionnez «Graph» et «View Dependency graph»: 8
Par défaut, le panneau du graphique des dépendances NDepend affiche le graphique des dépendances entre les assemblies .NET : 9
De nombreuses facilités sont offertes pour générer toutes sortes de graphes comme des graphes d'appels, des graphes de couplages entre deux composants, des graphes de portions de code trop monolithiques.

Conclusion

Je n'ai pas présenté toutes les fonctionnalités de NDepend car c'est un produit au spectre très large et complet. J'ai choisi les fonctionnalités les plus intéressantes selon moi.

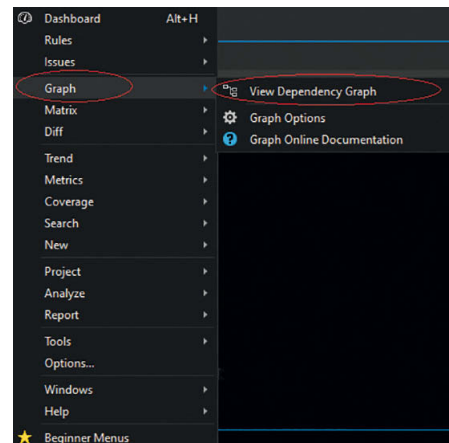
Je n'ai pas précisé que tous les résultats de NDepend peuvent être différenciés depuis une baseline : <https://www.ndepend.com/docs/code-diff-in-visual-studio>

Je n'ai pas exploré les capacités de la dette technique : en plus du graphe de dépendances, NDepend propose aussi une DSM (Dependency Structure Matrix <https://www.ndepend.com/docs/dependency-structure-matrix-dsm>) et des facilités pour générer des requêtes CQLinq portant sur les dépendances (<https://www.ndepend.com/docs/cqlinq-features#Dependencies>).

Visualisation des métriques de code avec un Treemap coloré : <https://www.ndepend.com/docs/treemap-visualization-of-code-metrics#Color>

En plus de la version développeur avec intégration dans Visual Studio, NDepend peut être intégré dans votre build process pour générer des rapports <https://www.ndepend.com/sample-reports/>. C'est la version Build-Machine. Une troisième version propose une intégration poussée dans Azure DevOps : <https://www.ndepend.com/docs/azure-devops-tfs-vsts-integration-ndepend>

Je pense honnêtement que NDepend est très puissant et très facile à utiliser. Le langage CQLINQ et le tableau de bord NDepend sont en fait les meilleures fonctionnalités de l'outil. Je n'ai rien trouvé d'équivalent à ce niveau. Cependant, parce que NDepend ne corrige pas le code pour vous, je suggère de continuer à uti-



8

liser Resharper qui peut fonctionner en synergie avec NDepend. Il y a peu de chevauchement entre les ensembles de règles NDepend et Resharper, tandis que Resharper se concentre davantage sur les détails à l'intérieur du corps des méthodes, NDepend se concentre davantage sur la structure du code et la qualité globale. L'équipe NDepend est très active et nous pouvons nous attendre à beaucoup de bons développements dans les futures versions.



Thibault Falque

Je m'appelle Thibault Falque, j'ai 24 ans. Je suis diplômé d'un master en informatique spécialisé en Intelligence Artificielle. Je réalise actuellement une thèse dans ce domaine et plus précisément dans le domaine de l'apprentissage et de la contrainte. Je réalise cette thèse chez Exakis Nelite en collaboration avec le laboratoire de recherche en informatique de Lens (CRIL).

Python 3.8 : quelles nouveautés ?

Chaque nouvelle version d'un langage ajoute de nouvelles fonctionnalités. Python 3.8 respecte la règle et introduit quelques nouveautés intéressantes. Je ne peux énumérer dans cet article l'ensemble des nouveautés, car il y en a beaucoup, je fais donc le point sur 3 fonctionnalités qui me semblent importantes. Un nouvel opérateur permettant d'affecter des variables dans une structure de contrôle, une nouvelle manière de gérer le positionnement des arguments dans une fonction et des ajouts dans la gestion des f-string.

Installation de Python 3.8

Pour Windows :

Pour les utilisateurs de Windows il suffit de télécharger la dernière version de l'exécutable depuis le site de Python (python.org) et de suivre les instructions.

Pour macOS :

Pour les utilisateurs de macOS, un installateur est également disponible depuis le site de Python, il suffira de l'installer comme vous avez l'habitude de faire avec d'autres logiciels. L'installateur est disponible pour macOS 10.9 (Mavericks) et version ultérieure, uniquement en version 64bits. Par défaut la version de Python installée sur macOS est la version 2.7.

Pour Linux (Ubuntu) :

Python est installé par défaut sur les systèmes Linux. Ubuntu 18.04 propose la version 2.7 et 3.6 par défaut. Pour installer la version 3.8 il faudra simplement faire `apt install python3.8`.

Migration de Python 2.X vers 3.X

Il n'est pas possible de mettre directement à jour une installation de Python 2.X vers une installation 3.X. Il faut nécessairement avoir deux environnements Python. C'est le cas par exemple sur Ubuntu 18.04 qui est fourni avec deux environnements Python par défaut. Pour migrer un projet s'exécutant sur Python 2.X (<2.7). Il faut suivre quelques recommandations afin de migrer le projet.

Tout d'abord et si c'est possible essayez de faire en sorte qu'il n'y est que la version 2.7 qui soit compatible pour votre projet. En effet les versions 2.7 et 3.X sont relativement compatibles. Si supprimer une compatibilité avec une version inférieure à 2.7 n'est pas possible le module `six` permet d'aider à rendre compatible un projet à la fois pour Python 2 et 3.

Il est important d'avoir un bon niveau de couverture de code pour ces tests unitaires, ainsi il sera plus simple de détecter une quelconque régression lors de la migration du code source. Utilisez des outils comme `futurize` ou `modernize` pour mettre à jour votre code. Utilisez `pylint` afin de détecter les mauvaises pratiques dans votre code. Il peut également être intéressant de savoir si les dépendances de votre projet sont compatible avec un passage vers Python 3.X. Pour cela il existe le module `caniusepython3` qui permet de vérifier la compatibilité. Enfin il existe un livre en libre accès

permettant de connaître les stratégies de migration vers Python et comment procéder (<http://python3porting.com/>).

Un nouvel opérateur qui fait du morse ...

La version 3.8 de Python introduit un tout nouvel opérateur: **the walrus operator**. C'est un opérateur permettant de faire une affectation dans une structure de contrôle (i.e if, while ...). Il est appelé ainsi pour sa ressemblance avec les yeux et les défenses d'un morse, il se note: `:=`. Dans les autres langages ce type d'affectation n'est pas nouveau mais cela l'est complètement pour Python. En effet il n'est pas rare de voir en C, le code suivant :

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4 int main()
5 {
6     if ((int) f = open()) != -1)
7     {
8         // Le fichier est ouvert nous pouvons l'utiliser
9     }
10    else
11    {
12        // Une erreur s'est produite lors de l'ouverture du fichier
13    }
14 }
```

En C l'affectation retourne la valeur affectée. Ici la variable `f` est affectée par le retour de l'appel système `open()`, la comparaison avec `-1` vérifie que l'appel système n'a pas échoué. Si tout s'est bien passé alors nous rentrons dans le `if`, dans le `else` dans le cas contraire.

```
1 if (n := len(a)) > 10:
2     print(f"List is too long ({n} elements, expected <= 10)")
```

L'utilisation de l'opérateur **walrus** dans la condition permet d'éviter un double appel à la fonction `len`. Notez ici l'utilisation des **f-string** apparues en **Python 3.6**.

On peut citer un autre exemple d'utilisation de ce nouvel opérateur lorsque l'on utilise des expressions régulières :

```
1 discount = 0.0
2 if (mo: = re.search(r'(\d+)% discount', advertisement)):
3     discount = float(mo.group(1)) / 100.0
```

Ici le résultat de la recherche est affecté à la variable `mo`, de plus le `if` permet de vérifier que l'expression régulière a réussi sur la chaîne de caractères `advertisement`. Avant ce type de code, nous l'aurions écrit de la manière suivante :

```

discount = 0.0
mo = re.search(r'(\d+)% discount', advertisement)
if mo:
    discount = float(mo.group(1)) / 100.0

```

Fondamentalement, l'opérateur n'apporte pas, dans ce cas, un gain énorme, mais peut se révéler néanmoins utile. Un autre exemple est dans les boucles *for* sur une ligne.

```

[clean_name.title() for name in names
 if (clean_name:=normalize('NFC', name)) in allowed_names]

```

L'opérateur **walrus** permet d'éviter un double appel à la fonction *normalize*. Le code en Python avant l'introduction de l'opérateur aurait été :

```

[clean_name.title() for name in names if (clean_name:=normalize('NFC', name)) in allowed_names]

```

Ce nouvel opérateur peut s'avérer pratique dans beaucoup de cas, mais attention, un usage abusif peut nuire à la lisibilité du code.

Paramètres positionnels uniquement

Jusqu'à maintenant en python il était possible de forcer des arguments d'un appel de fonction à être appelé en mode *keyword* (i.e *func(a=10)*). Il est maintenant possible de forcer les arguments à être passés en mode *positionnel* (i.e *func(10)*). Voyons comment :

```

def to_string_user(*, name=None, first_name=""):
    print(f"{name=} {first_name=}")

```

Dans l'exemple précédent les arguments de la fonction ne peuvent être passés qu'en mode *keyword*, grâce au symbole ***. Ainsi l'appel de fonction suivant échouera :

```
to_string_user("Dupont", "Jean")
```

Il est maintenant possible de forcer des arguments positionnels avec le symbole *.*.

```

def to_string_user(/, name, first_name):
    print(f"{name=} {first_name=}")

```

L'appel à la fonction suivant échouera :

```
to_string_user(name="Dupont", first_name="Jean")
```

La documentation officielle de Python indique des cas d'usages intéressants, pour cette fonctionnalité, comme par exemple pour la fonction intégrée *len* dont la signature est *len(obj, /)*. Cette fonction force l'utilisation des arguments positionnels, car un appel à *len* en utilisant le *keyword* *obj* dans l'appel de la fonction n'est pas clair et réduit la lisibilité.

De plus, prenons le cas où vous utilisez une bibliothèque. Vous allez forcément utiliser des fonctions ou des méthodes de cette bibliothèque. Si l'auteur force certaines de ces fonctions (ou méthodes) à utiliser des arguments *positionnels*, alors il pourra les renommer sans risque de casser votre utilisation.

```

def quantiles(dist, /, *, n=4, method='exclusive'):
    ...

```

Cette fonction peut aisément renommer son premier paramètre *dist* sans risque. Le fait de forcer l'utilisation d'un argument *positionnel* permet d'éviter l'utilisation d'un argument *keyword* et donc que le code ne fonctionne plus si le nom de l'argument change.

Support du = dans les f-string

Dans Python 3.6 la fonctionnalité **f-string** arrivait et permettait d'écrire le code **magnifique** suivant :

```

first_name = "Jean"
name = "Dupont"
number = 10
print(f'{first_name} {name} a {number} ans.')

```

Python 3.8 va plus loin en proposant d'ajouter le symbole égal dans les accolades et ainsi automatiquement écrire dans la chaîne *nom_var=valeur_var*.

```

first_name = "Jean"
name = "Dupont"
number = 10
print(f'{first_name=} {name=} {number=}')

```

La sortie de ce code sera :

```
first_name=Jean name=Dupont number=10
```

Personnellement, j'utilise beaucoup cette fonctionnalité à des fins de débogage.

Conclusion

Cette nouvelle version de Python apporte quelques nouveautés intéressantes. Certaines ne doivent pas être utilisées trop fréquemment car elles peuvent nuire à la lisibilité mais elles sont dans la plupart des cas utiles. La documentation officielle de Python énumère beaucoup d'autres nouvelles fonctionnalités ou ajouts. Il ne faut pas hésiter à aller la regarder.

Documentation officielle : <https://docs.python.org/3/whatsnew/3.8.html>



Philippe BOULANGER
Manager des expertises C/C++ et Python
www.invivoo.com



Quelques nouveautés en Python 3.8

Python 3.8 nous arrive avec de nombreuses nouveautés : certaines dans l'API C, d'autres dans la syntaxe du langage ou dans de nouvelles fonctionnalités. Dans cet article nous laisserons de côté de nombreux ajouts et corrections comme par exemple l'API C, l'opérateur morse (walrus en anglais : `=`), les arguments positionnels ainsi que les changements sur les F-Strings.

Bloc « finally »

Jusqu'à présent, du fait d'une erreur d'implémentation, il n'était pas possible d'utiliser le mot-clé `continue` dans un bloc `finally`. C'est désormais corrigé !

Généralisation de « `as_integer_ratio` »

Cette fonction était disponible sur les types `float` et `decimal.Decimal`. Elle a, désormais, été rajoutée aux types numériques (`bool`, `int` et `fractions.Fraction`). Cette généralisation va permettre d'utiliser la séquence de code suivante sans se poser de question sur le type manipulé :

```
numerator, denominator = x.as_integer_ratio()
```

Expressions régulières

Le module `re` a reçu une amélioration minime qui impacte la définition des expressions régulières qui contiennent des caractères Unicode. On va désormais pouvoir les nommer explicitement :

```
notice = 'Copyright © 2019'
copyright_year_pattern = re.compile(r'\W{copyright sign}\s*(\d{4})')
print(int(copyright_year_pattern.search(notice).group(1)))
```

Le caractère © est représenté par le groupe `\W{copyright sign}`. Où « copyright sign » est le nom du caractère. C'est bien plus lisible pour un développeur qui lit le code que d'utiliser `\u00A9`...

L'ordre d'insertion dans les dictionnaires

Dans une version précédente de Python, la fonctionnalité de conservation de l'ordre d'insertion des couples (clé, valeur) a été ajoutée dans les dictionnaires dict. La version 3.8 de Python nous rajoute l'implémentation de la fonction `reversed`. Cela permet d'itérer dans le sens inverse de l'insertion :

- sur les couples (clé, valeur)
- sur les clés
- sur les valeurs.

Voici quelques exemples d'utilisation tirés de la description de l'anomalie :

```
>>> d = dict(a=1, b=2)
>>> list(reversed(d))
['b', 'a']
>>> list(reversed(d.keys()))
['b', 'a']
>>> list(reversed(d.values()))
```

```
[2, 1]
>>> list(reversed(d.items()))
[('b', 2), ('a', 1)]
```

Unpacking

Empaqueter et dépaqueter les données (tableau et tuples) sont des actions qui permettent de simplifier le code. Voici un exemple de code en Python 3.7 :

```
def parse(text):
    t, *other = text.split()
    return (t.upper(), *other)

print(parse("toto tata titi"))
```

En Python 3.7, nous étions obligés d'utiliser des parenthèses pour retourner un tuple dans la fonction `parse`. EN Python 3.8, nous allons pouvoir économiser quelques caractères :

```
def parse(text):
    t, *other = text.split()
    return t.upper(), *other

print(parse("toto tata titi"))
```

Dictionnaires par compréhension

Une incohérence entre les dictionnaires littéraux et les dictionnaires par compréhension existait dans les versions de Python 3.7 et antérieures. Dans un dictionnaire littéral la clé est évaluée avant la valeur :

```
>>> {input("key?"):input("value?")}

key?a

value?1
{'a': '1'}
```

Alors que pour les dictionnaires par compréhension c'est l'inverse :

```
>>> {input("key?"):input("value?") for i in range(2)}

value?1

key?a
```

Abonnez-vous à **Programmez!** Abonnez-vous à **Programmez!** Abonnez-vous à **Programmez!**

PROGRAMMEZ!
Le magazine des développeurs

Offres printemps **2020**

Nos classiques

1 an
11 numéros **49€***

2 ans
22 numéros **79€***

Etudiant
1 an - 11 numéros **39€***

* Tarifs France métropolitaine

Abonnement numérique

PDF **35€**
1 an - 11 numéros

Option : accès aux archives **15€**

Souscription uniquement sur
www.programmez.com

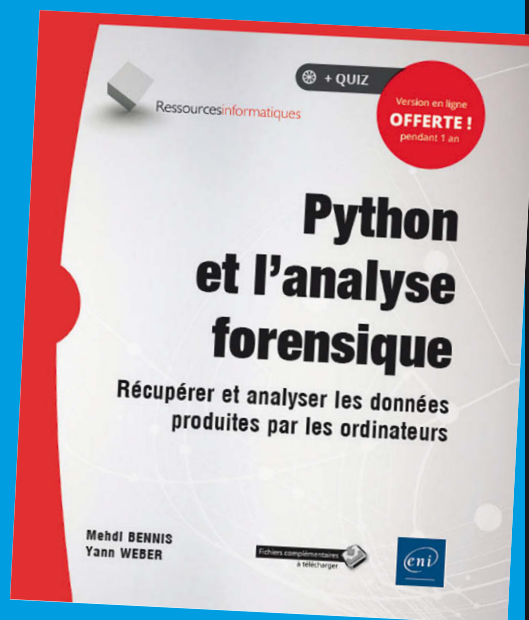
1 an
11 numéros
+ Python et
l'analyse forensique
(éditions ENI)

50€*

2 ans
22 numéros
+ Python et
l'analyse forensique
(éditions ENI)

80€*

* Offres limitées à la France Métropolitaine



Toutes nos offres sur www.programmez.com



Oui, je m'abonne

- ☐ **Abonnement 1 an** : 49 €
- ☐ **Abonnement 2 ans** : 79 €
- ☐ **Abonnement 1 an Etudiant** : 39 €

Photocopie de la carte d'étudiant à joindre

OFFRES PINTemps 2020

- ☐ **Abonnement 1 an** : 50 €
- ☐ **Abonnement 2 ans** : 80 €

☐ Mme ☐ M. Entreprise : _____ Fonction : _____

Prénom : _____ Nom : _____

Adresse : _____

Code postal : _____ Ville : _____

email indispensable pour l'envoi d'informations relatives à votre abonnement

E-mail : _____ @ _____

☐ Je joins mon règlement par chèque à l'ordre de Programmez !

☐ Je souhaite régler à réception de facture

* Tarifs France métropolitaine

Abonnez-vous à **Programmez!** Abonnez-vous à **Programmez!** Abonnez-vous à

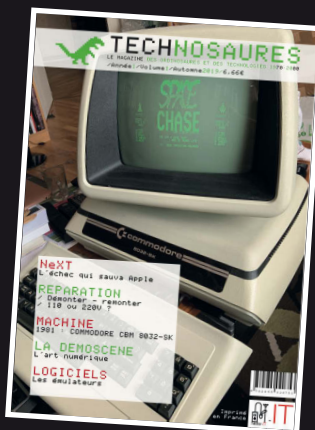
NOUVEAU ! Boutique Programmez!

Les anciens numéros disponibles

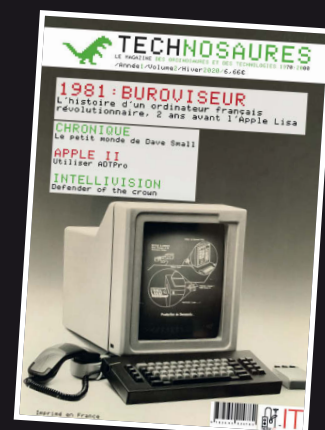


Technosaures

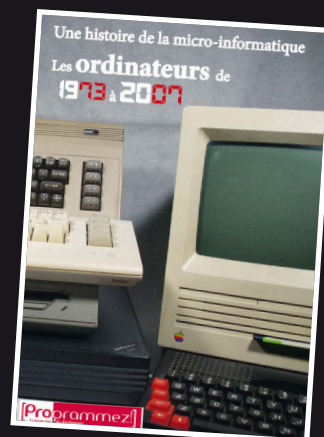
Prix unitaire : **7,66 €**
(frais postaux inclus)



n°1



n°2



Histoire de la
micro-informatique
1973-2007

12,99 €
(frais postaux inclus)

tarif unitaire 6,5 € (frais postaux inclus)

- | | | | | | |
|------------------------------|---|-----------------------------|------------------------------|---|-----------------------------|
| <input type="checkbox"/> 226 | : | <input type="checkbox"/> ex | <input type="checkbox"/> 237 | : | <input type="checkbox"/> ex |
| <input type="checkbox"/> 229 | : | <input type="checkbox"/> ex | <input type="checkbox"/> 238 | : | <input type="checkbox"/> ex |
| <input type="checkbox"/> 233 | : | <input type="checkbox"/> ex | <input type="checkbox"/> 239 | : | <input type="checkbox"/> ex |
| <input type="checkbox"/> 235 | : | <input type="checkbox"/> ex | | | |

Technosaures ☐ N°1 ☐ N°2
soit exemplaires x 7,66 € = €
☐ Histoire de la Micro-informatique 12,99 €

soit exemplaires x 6,50 € = € soit au **TOTAL** = €

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

E-mail : @

Règlement par chèque à l'ordre de Programmez ! | Disponible sur www.programmez.com


```
value?2
key?b
{'a': '1', 'b': '2'}
```

Cette incohérence est désormais corrigée et la clé est toujours évaluée avant la valeur.

Module pickle

Le module pickle a reçu une grosse modification de son protocole de sauvegarde : le protocole 5. Si elle est transparente pour l'utilisateur en termes de code, elle permet de tirer parti des processeurs multicœurs pour optimiser les transferts en minimisant les copies mémoires et potentiellement en compressant certaines données. Cela permet d'améliorer la gestion de données ayant une forte volumétrie.

Module importlib.metadata

Ce module a été ajouté afin de permettre d'accéder aux métadonnées des modules externes. On peut accéder :

- à la version
- à la liste des points d'entrées
- à la liste des fichiers installés pour le module/package
- à la liste des packages requis pour un module donné.

Module asyncio

Le module asyncio rajoute une fonction simplificatrice. En 3.7 nous étions obligés d'écrire :

```
import asyncio

async def main():
    await asyncio.sleep(0)
    return 42

loop = asyncio.new_event_loop()
```

```
asyncio.set_event_loop(loop)
try:
    loop.run_until_complete(main())
finally:
    asyncio.set_event_loop(None)
    loop.close()
```

En Python 3.8 nous nous contenterons de :

```
import asyncio
async def main():
    await asyncio.sleep(0)
    return 42

asyncio.run(main())
```

De même, nous pouvons désormais lancer l'interpréteur via la commande « python -m asyncio » pour exécuter une commande await de haut niveau. Cela simplifie les tests, car on peut exécuter directement :

```
import asyncio
await asyncio.sleep(10, result='hello')
```

Module cProfile

La classe cProfile.Profile peut désormais être utilisée via un gestionnaire de contexte (mot-clé `with`).

Module csv

La classe csv.DictReader retourne une instance de `dict` ce qui permet de conserver l'ordre des clés tout en économisant la mémoire consommée et en améliorant les performances.

Conclusion

Couvrir toutes les modifications qui ont été apportées entre les versions 3.7 et 3.8 est une tâche trop longue pour être couverte par un seul article. De nombreux autres modules ont été amendés...



1 an de Programmez!

ABONNEMENT PDF : 35 €

Abonnez-vous sur :
www.programmez.com



Mathias Deremer-Accettone
Consultant technique à Ineat
<https://blog.ineat-group.com/>

Monitoring Java sans effort avec InspectIt Ocelot

Garder un œil sur l'activité des applications que nous développons est aujourd'hui un élément clé de tout SI qui se respecte. Les solutions disponibles sur le marché sont nombreuses. Pourtant, exposer et traiter des métriques peut rapidement devenir un véritable casse-tête. Surtout quand il s'agit de rendre observable une application existante. Mais ne vous en faites pas, InspectIt Ocelot vole au secours de vos projets Java !

Agent Ocelot, à votre service.

Lorsqu'on souhaite mettre en place du monitoring sur un projet, deux solutions s'offrent à nous :

- Utiliser un APM (Application Performance Management) comme AppDynamics ou Dynatrace pour observer nos applications.
- S'appuyer sur des solutions comme Micrometer pour exposer des métriques, qui seront ensuite affichées dans des dashboards (type Grafana).

Les deux approches ont leurs forces, mais également leurs faiblesses : la première a un coût d'entrée généralement élevé (coûts de licence importants), et la seconde implique de modifier le code source de l'application... Tâche qui peut devenir longue et fastidieuse si ce facteur n'a pas été pris en compte dès le début des développements. Mais une autre approche est possible... approche à laquelle on ne pense pas toujours : l'utilisation d'agents paramétrables scrutant l'activité de la JVM afin d'en extraire des indicateurs relatifs à l'activité d'une application Java. Et justement InspectIt a open-sourcé un agent de ce type il y a quelques mois ! Son nom : Ocelot. Son utilisation a pour principal avantage de permettre le contrôle des performances d'une application dont nous ne pouvons pas modifier le code source. Sans proposer l'intégralité des fonctionnalités d'un APM ni d'interface graphique, Ocelot peut tout de même :

- s'intégrer facilement avec bon nombre de technologies (Jaeger, Prometheus, Influx, Grafite, ...);
- découvrir automatiquement les dépendances entre microservices;
- prendre à chaud les changements de configuration, sans redémarrage nécessaire de l'application monitorée ou de l'agent lui-même;
- réaliser du tracing distribué.

Prise en main

Premier démarrage

L'agent est téléchargeable via la commande suivante :

```
wget https://github.com/inspectIT/inspectit-ocelot/releases/download/0.4/inspectit-ocelot-agent-0.4.jar
```

Dès lors, il est possible de démarrer conjointement agent et application monitorée avec :

```
java -javaagent:"/path/inspectit-ocelot-agent-0.4.jar" -jar app.jar
```

Nous aurions pu également attacher l'agent à une application en cours d'exécution comme suit :

```
java -jar inspectit-ocelot-agent-0.4.jar [PID] '{inspectit:{service-name:"service name"}}'
```

où [PID] serait remplacé avec le Process ID de l'application.

Dans le cas présent l'agent scrutera l'activité de l'application « app.jar », et remontera uniquement les métriques standard.

Les sources de configuration

Cependant InspectIt Ocelot ne se limite pas à cela, et il est possible de paramétrer d'autres indicateurs que ceux mis à disposition par défaut. Ce paramétrage peut être réalisé de plusieurs façons. En outre il est possible de passer une configuration :

- au lancement de l'agent, en passant simplement la configuration en argument sous la forme d'une chaîne de caractères.

```
java -javaagent:/path/inspectit-ocelot-agent-0.4.jar="{\"inspectit\":{\"service-name\":\"monitored service\"}}\" -jar app.jar
```

- en créant une variable d'environnement

```
java -Dinspectit.service-name="monitored service" -javaagent:/path/inspectit-ocelot-agent-0.4.jar -jar app.jar
```

- en externalisant les configurations dans un fichier Yaml, et en le passant en option à l'agent (c'est généralement la solution retenue)

```
java -Dinspectit.config.file-based.path="/path/config.yaml" -javaagent:/path/inspectit-ocelot-agent-0.4.jar -jar app.jar
```

Sachez que pour ce dernier cas de figure les modifications effectuées sur config.yaml peuvent être reprises à chaud, en précisant une fréquence de rafraîchissement avec les options -Dinspectit.config.file-based.watch et -Dinspectit.config.file-based.frequency.

En pratique cela pourrait donner par exemple :

```
java -Dinspectit.config.file-based.path="/path/config.yaml" -Dinspectit.config.file-based.watch=true -Dinspectit.config.file-based.frequency=10s -javaagent:/path/inspectit-ocelot-agent-0.4.jar -jar app.jar
```

Personnaliser les métriques

Nous venons de voir qu'un agent lié à une JVM pouvait s'appuyer sur différentes sources de configurations, et notamment sur des fichiers Yaml. Ces configurations sont composées de quatre sections : « actions », « metrics », « scopes » et « rules ». Pour illustrer le fonctionnement de chaque section, partons du principe que nous avons à disposition une application écrite en Java, et que nous souhaitons mesurer le temps de traitement d'une méthode nommée « handle » (définie dans une classe « AdderHandler »), ainsi que le nombre d'appels à celle-ci. Voyons comment paramétrer chaque section.

Actions

Les « actions » peuvent être vues comme des fonctions à appliquer sur les données mesurées (les traitements réalisés par ces actions pouvant être définis en Java).

```
actions:
```

```
# HELP method_duration Processing duration from method entry to method exit
# TYPE method_duration gauge
method_duration{get_method_full_name="AdderHandler.handle",host="ff89c5413739",host_address="172.22.0.4",service="adder-service",} 0.193577
# HELP method_duration_count Total calls number to the method
# TYPE method_duration_count counter
method_duration_count{get_method_full_name="AdderHandler.handle",host="ff89c5413739",host_address="172.22.0.4",method_name="",service="adder-service",} 32.0
# HELP http_out_responsetime_sum sum of http/out/responsetime [ms]
# TYPE http_out_responsetime_sum counter
# HELP http_out_count count of http/out/responsetime [ms]
# TYPE http_out_count counter
# HELP service_in_count count of service/in/responsetime [ms]
# TYPE service_in_count counter
# HELP service_in_responsetime_sum sum of service/in/responsetime [ms]
# TYPE service_in_responsetime_sum counter
# HELP http_in_responsetime_sum sum of http/in/responsetime [ms]
# TYPE http_in_responsetime_sum counter
# HELP http_in_count count of http/in/responsetime [ms]
# TYPE http_in_count counter
# HELP jvm_gc_Concurrent_phase time sum of the time spent in concurrent GC phases [ms]
# TYPE jvm_gc_Concurrent_phase time counter
# HELP jvm_gc_memory_allocated sum of the increase in the size of the young generation memory pool after one GC to before the next [bytes]
# TYPE jvm_gc_memory_allocated counter
jvm_gc_memory_allocated{host="ff89c5413739",host_address="172.22.0.4",service="adder-service",} 6.89871192E8
```

1

```
get_method_full_name:
  input:
    _methodName: String
    _class: Class
  value: "new
StringBuilder(_class.getSimpleName()).append('.').append(_methodName).toString()"
```

Une seule action est définie dans le bloc Yaml précédent. « get_method_full_name » prend deux paramètres en entrée : le nom de la méthode ainsi que la classe sur laquelle l'action sera appliquée. Ces paramètres sont spécifiés dans la sous-section « input ». « value » indique la valeur renvoyée par l'action. Dans notre cas il s'agit de la concaténation du nom de la classe avec celui de la méthode.

Metrics

C'est dans cette section que sont définies les métriques, c'est-à-dire les points de mesure. Par exemple le temps de traitement d'un bloc de code ou nombre d'appels à telle ou telle méthode.

```
metrics:
  definitions:
    '[method/duration]':
      unit: ms
      views:
        '[method/duration]':
          aggregation: LAST_VALUE
          description: "Processing duration from method entry to method exit"
          tags:
            get_method_full_name: true
        '[method/duration/count]':
          aggregation: COUNT
          description: "Total calls number to the method"
          tags:
            get_method_full_name: true
```

Dans l'exemple précédent deux points de mesure sont définis :

- [method/duration] qui permettra de mesurer le temps de traitement. Grâce au mot clé « aggregation » nous spécifions ici que c'est la dernière valeur mesurée qui sera retournée.
- [method/duration/count] qui totalisera le nombre d'appels à tel ou tel bloc de code. Contrairement au point précédent, nous souhaitons obtenir un cumul et pour cela nous positionnons l'« aggregation » à « COUNT ».

Les « tags » permettent de spécifier les actions à appliquer sur nos métriques.

Scopes

Les « scopes » permettent de définir le périmètre d'application des « actions ». Nous pouvons donc « demander » à l'agent d'effectuer des mesures sur des blocs de code bien précis.

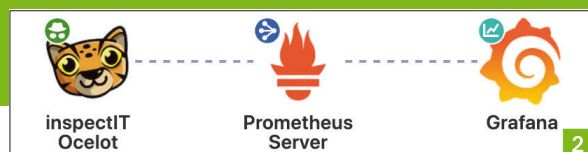
```
scopes:
  rest-call-scope:
    type:
      name: "com.lineat.inspectit.demo.AdderHandler"
      matcher-mode: "EQUALS_FULLY"
    methods:
      - name: "handle"
        matcher-mode: "EQUALS_FULLY"
    visibility: [PUBLIC]
```

Le scope « rest-call-scope » s'appliquera ici sur la méthode « handle » de la classe « AdderHandler » (spécifié via le mot clé « type »). La recherche du nom de classe et de la méthode est effectuée de façon stricte. Il est toutefois possible de rendre la recherche plus permissive pour par exemple « observer » toutes les méthodes commençant par un pattern particulier (dans ce cas de figure, nous positionnerions le matcher_mode à « START_WITH »).

Rules

Les « rules » précisent de quelle façon les « metrics » vont être extraites et traitées. En outre, les « rules » permettent de lier « actions », « scopes » et « metrics ».

```
rules:
  rest_call_duration:
    scopes:
      rest-call-scope: true
    entry:
      method_entry_time:
        action: timestamp_nanos
      get_method_full_name:
        action: get_method_full_name
    exit:
      method_duration:
        action: elapsed_millis
      data-input:
        sinceNanos: method_entry_time
    metrics:
      '[method/duration]': method_duration
```



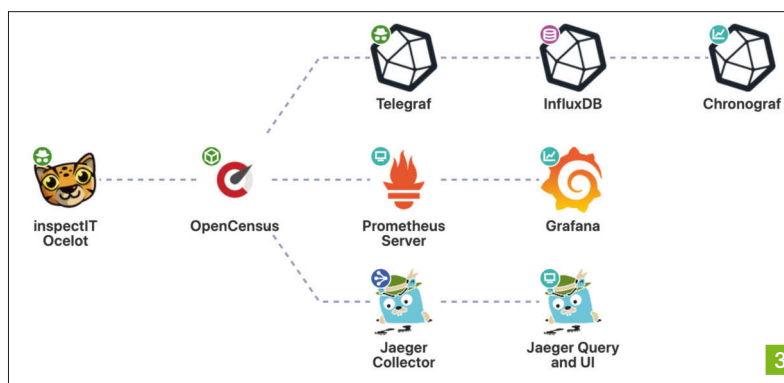
À titre indicatif, voici le code complet du fichier Yaml contenant la configuration que nous venons de définir :

```

inspectit:
instrumentation:
actions:
get_method_full_name:
input:
_methodName: String
_class: Class
value: "new
StringBuilder(_class.getSimpleName()).append('.').append(_methodName).toString()"
scopes:
rest-call-scope:
type:
name: "com.ineat.inspectit.demo.AdderHandler"
matcher-mode: "EQUALS_FULLY"
methods:
- name: "handle"
matcher-mode: "EQUALS_FULLY"
visibility: [PUBLIC]
rules:
rest_call_duration:
scopes:
rest-call-scope: true
entry:
method_entry_time:
action: timestamp_nanos
get_method_full_name:
action: get_method_full_name
exit:
method_duration:
action: elapsed_millis
data-input:
sinceNanos: method_entry_time
metrics:
'[method/duration]': method_duration
metrics:
definitions:
'[method/duration]':
unit: ms
views:
'[method/duration]':
aggregation: LAST_VALUE
description: "Processing duration from method entry to method exit"
tags:
get_method_full_name: true
'[method/duration/count]':
aggregation: COUNT
description: "Total calls number to the method"
tags:
get_method_full_name: true
  
```

Exporter les métriques

Une fois le paramétrage réalisé dans le fichier Yaml, il ne reste plus qu'à lancer l'agent et l'application.



```
java -Dinspectit.config.file-based.path="/path/config.yaml" -javaagent:/path/inspectit-ocelot-agent-0.4.jar -jar app.jar
```

Dès lors les métriques sont accessibles sur <http://localhost:8888> (si les opérations sont réalisées en local). **1**

À ce stade les informations dont nous avons besoin sont disponibles, mais difficilement compréhensibles. Ocelot met à disposition des Exporters, permettant de définir dans quel format les données doivent être exportées. L'exporter Prometheus est celui utilisé par défaut par Ocelot, ainsi pour exposer les données à Prometheus il suffit de préciser le port et l'adresse de ce dernier. **2**

En local cela pourrait donner :

```
java -Dinspectit.exporters.metrics.prometheus.host="0.0.0.0" -Dinspectit.exporters.metrics.prometheus.port="8088" -javaagent:/path/inspectit-ocelot-agent-0.4.jar -jar app.jar
```

Mais Ocelot propose d'autres types d'exporters, et notamment un exporter pouvant « dialoguer » avec un agent Opencensus et ainsi produire des données génériques destinées à d'autres briques comme Jaeger, Telegraf, Zipkin ... **3**

Son utilisation se traduirait alors par :

```
java -Dinspectit.exporters.metrics.opencensus-agent.enabled=true -Dinspectit.exporters.metrics.open-census-agent.address="localhost:1234" -javaagent:/path/inspectit-ocelot-agent-0.4.jar -jar app.jar
```

Il ne reste plus qu'à référencer les endpoints exposant vos métriques dans Prometheus ou toute autre solution de monitoring.

Conclusion

Et voilà comment monitorer efficacement une application Java sans toucher au code source, avec simplement quelques lignes de Yaml ! Les différents mots clés utilisés dans la configuration Yaml pourront peut-être en déstabiliser certains d'entre vous, mais avec un peu d'entraînement vous pourrez très vite mettre en place des stacks de monitoring complètes basées sur Ocelot. Sachez enfin que nous n'avons qu'effleuré le potentiel de cette technologie et qu'elle ne se limite pas uniquement aux relevés de métriques.

Liens utiles

Site officiel : <https://www.inspectit.rocks/>

Ocelot en pratique : <https://blog.ineat-group.com/2019/10/monitoring-java-sans-effort-avec-inspectit-ocelot/>

OpenApm : <https://openapm.io/landscape>

OpenCensus : <https://opencensus.io/>

Prometheus + Grafana : <https://prometheus.io/docs/visualization/grafana/>



Andréas HANSS

Expert en développement JavaScript chez Kaliop
 Tech Lead JavaScript Cross-Platform et j'ai — au moment où j'écris ces lignes — 3 ans d'expérience avec React Native. Je viens du web avec ReactJS / NodeJS, mon choix préféré est donc naturellement React Native (nous verrons pourquoi). Mais j'aime aussi beaucoup Flutter. Je suis son évolution de près et je suis sûr que le framework à un bel avenir.

React Native vs Flutter : comparaison d'un point de vue d'une vraie expérience projet

Avec quel outil commencer le développement multi-plateforme ? Cette comparaison construite autour de retours d'expérience concrets va vous aider à faire le bon choix entre React Native et Flutter.

Vous en avez certainement entendu parler car ils font le buzz. Ces deux outils révolutionnent le développement d'applications mobiles et le rendent plus accessible à de nombreux développeurs, d'où les nombreuses comparaisons apparues au cours des derniers mois. Cependant, tout le monde parle d'arguments qui n'ont pas vraiment d'importance comme le langage ou l'outil de ligne de commande pour l'installation. J'aimerais partager avec vous une approche différente en parlant des choses qui comptent réellement dans un projet, afin de choisir le cadre approprié à vos besoins.

Tout dépend de vous : d'où vous venez et de vos besoins. Les deux sont d'excellents outils que vous devriez considérer pour vos applications.

Maintenant que tout a été dit, nous pouvons commencer.

Prise en main

Il y a déjà beaucoup de comparaisons entre Dart et JS, composants natifs, scaffolding, etc. Je ne développerai pas plus cela ici. N'hésitez pas à faire des recherches sur Internet et vous obtiendrez de nombreux résultats pertinents, notamment des benchmarks de performances.

En bref : Dart est nativement typé. C'est un très bon langage qui a été conçu à l'origine pour remplacer JavaScript (mais Google n'a pas réussi à intégrer dans d'autres navigateurs que Chrome une machine virtuelle Dart malgré plusieurs tentatives).

D'autre part, JavaScript ne possède pas nativement de types statiques mais peut

être étendu avec TypeScript notamment pour ajouter cette fonctionnalité.

On pourrait parler de votre IDE favori ou de bien d'autres outils de développement...

Mais ce n'est pas ce qui compte vraiment.

Nous nous soucions de la liberté et de la flexibilité qu'un framework nous donne pour atteindre nos objectifs. Non pas de la façon dont il est pris en main à première vue. On s'habitue à tout après tout, non ? Surtout quand on est développeur et que les technologies évoluent vite.

Parlons donc des arguments populaires que l'on peut lire partout, les mêmes arguments que ceux que j'entends de la bouche de mes collègues au travail.

La mauvaise argumentation entre Flutter et REACT NATIVE

« Dart est plus sûr car c'est un langage à types statiques alors que JS ne l'est pas. »

Le typage est un choix de premier ordre pour n'importe quelle application, en particulier pour les applications mobiles qui ont une plus grande inertie lors du déploiement de mises à jour. Ceci notamment à cause du processus de build et de la validation des stores applicatifs.

JS peut utiliser TypeScript et Flow pour également obtenir du typage statique. Cela ne donne pas encore d'optimisation de performance comme Dart le fait actuellement. Cela pourrait toutefois changer à l'avenir, en particulier avec le nouveau moteur/architecture de REACT NATIVE qui sera disponible courant 2020 selon les prévisions.

Malgré cela, il peut parfois être intéressant

d'avoir un langage non-compilé. Par exemple pour un procédé de mise à jour à distance, dont nous parlerons plus loin.

« Flutter et Dart sont propulsés par Google et le nouveau système d'exploitation Google Fuchsia est l'avenir. REACT NATIVE mourra en douceur car Flutter sera beaucoup mieux optimisé pour son nouveau système d'exploitation. »

Personne ne peut prédire l'avenir, mais une chose est certaine : quelques-unes des plus grandes entreprises du monde sont en train de parier sur React Native en ce moment.

Microsoft crée de nombreux outils tels que CodePush ou REACT NATIVE Windows / Mac et une partie de leurs outils populaires tels que Word sont portés partiellement en React Native.

On parle également d'entreprises et start-ups telles que Discord, Tesla, Uber ou encore Pinterest et bien évidemment Facebook et Instagram.

Flutter est certainement un choix fort et continuera à se développer, mais je pense que les deux frameworks resteront des concurrents pour Fuchsia.

La bonne argumentation entre React Native et Flutter

« Les performances de Flutter sont meilleures que celle de REACT NATIVE. »

Effectivement, grâce à la compilation en amont c'est complètement vrai. Cependant il faut noter qu'avec quelques optimisations, on peut obtenir une expérience com-

plètement fluide et performante avec React Native également.

« **Quand on vient du monde Android/iOS, il est plus facile de prendre en main Flutter que REACT NATIVE.** »

En effet, Flutter est clairement le successeur de Java et Android et par les caractéristiques de Dart cela est plus simple à prendre en main quand on vient du natif.

Une bonne partie de la logique vient de Java et d'un langage objet fortement typé. La plus grosse différence se trouve dans l'apprentissage du côté déclaratif et l'organisation en Widgets, dont les développeurs natifs n'ont pas forcément l'habitude.

Les Widgets sont des composants métiers/graphiques avec des paramètres qui décrivent l'état et la mise en page escomptée de l'application.

« **Les outils de développement de Flutter sont bien mieux que ceux de REACT NATIVE.** »

Il y a en effet un grand écart entre les deux produits sur ce point. L'équipe derrière Flutter fournit une excellente intégration pour **VScode** et **Android Studio** avec de nombreux outils d'analyse et de debug.

De l'autre côté, chez REACT NATIVE, les outils sont relativement légers. Mis à part le React Dev Tool et quelques plugins, la plupart du debugging est délégué aux outils natifs d'Android et iOS. Facebook essaye d'améliorer cela, notamment grâce à leur nouvel outil **Flipper**, mais cela reste encore assez expérimental.

Comme nous pouvons le voir, il y a de bons arguments en faveur de nos deux frameworks. Allons plus loin dans les différences majeures en termes de styles et de comportements.

Les différences majeures entre REACT NATIVE et Flutter

Le moteur de rendu ne fonctionne pas du tout pareil !

Sur React Native, le JavaScript pilote à travers ce qu'on appelle le pont natif (native bridge), le moteur de rendu de chaque sys-

tème d'exploitation mobile. Il y a donc une sorte de **bus de message** entre les deux threads (JS et natif).

De ce fait les composants affichés sont ceux propres à chaque plateforme. On peut parfois être contraint d'utiliser des structures conditionnelles pour utiliser des composants natifs selon la plateforme, ce qui reste rare globalement.

Le rendu est de temps en temps légèrement différent sur les deux plateformes. Ces différences peuvent porter atteinte au fonctionnement de quelque chose si l'on ne fait pas attention. Il faut donc bien vérifier les deux plateformes lors du développement.

Chez Flutter, les applications sont rendues dans une sorte de **Canva**. De la même manière qu'un jeu vidéo se rend via son moteur graphique. Du coup, peu importe la plateforme, le rendu est exactement le même. Ce que fait Flutter, c'est qu'il redessine des composants reproduisant le même design que les composants natifs selon le design kit choisi. Cela laisse moins de place aux incompatibilités et cas à la marge.

Par contre, pour le driver web, Flutter utilise des canvas HTML. Ceci ampute aujourd'hui le SEO d'applications web en Flutter (actuellement en bêta), là où react-native-web rend de vraies balises HTML.

Mises à jour à la volée et compilation

Grâce à la nature de JavaScript qui est d'être compilé Juste-à-Temps, il est possible de tirer parti des mises à jour à la volée avec React Native. Et c'est vraiment très simple en utilisant Microsoft **Codepush** ou **Expo.io**. Une mise à jour à-la-volée est une mise à jour que l'on peut envoyer via le réseau, très rapidement sans build, ni validation de stores applicatifs. Cela sur tous les terminaux utilisateurs sans attendre qu'ils déclenchent une mise à jour.

C'est actuellement la principale raison pour laquelle je ne change pas REACT NATIVE pour Flutter progressivement (outre le fait que j'aime vraiment le paradigme React). Cela change vraiment la donne, en particulier sur les applications événementielles où le délai de mise à jour est très court et où vous avez beaucoup d'utilisateurs dans un court laps de temps.

Avec Expo.io, on peut même développer une application mobile sans environnement de développement installé sur son ordinateur.

Chez Flutter, ce n'est pas possible actuellement. Peut-être que l'équipe de Flutter nous fournira une meilleure solution à l'avenir, mais à l'heure actuelle, il n'y a pas de bibliothèque officielle solide qui supporte cela.

L'expérience de développement et la mise en page Outils de développement

Les outils de débogage sont géniaux avec Flutter et c'est un argument fort en faveur de l'expérience de développement. Vous pouvez aller plus facilement et plus profondément dans l'optimisation que dans REACT NATIVE comme nous l'avons vu plus haut dans cet article.

Définition de l'interface utilisateur

Mais d'un autre côté, alors que React native s'appuie sur **JSX** (une sorte de XML) pour décrire la définition de l'interface utilisateur de manière simple, Flutter utilise une mise en page basée sur des **Widgets**, ce qui reste assez verbeux et complique le développement selon moi.

Il y a beaucoup de répétitions et sans IDE, il devient vite difficile de lire le code. Cela complique évidemment les revues de **Pull Request**.

JSX tire parti d'une définition de mise en page simple et lisible, proche du HTML, encore plus avec une bibliothèque telle que **styled-components**. Cela donne à REACT NATIVE un bel avantage, car n'importe quel développeur venant du web peut rapidement commencer avec REACT NATIVE.

La communauté et les librairies

L'écosystème JavaScript est très populaire avec beaucoup d'outils et de bibliothèques, cependant bien que Dart soit un bel outil, la communauté est aujourd'hui beaucoup plus petite et beaucoup plus jeune.

Vous avez beaucoup de paquets avec Dart, mais la plupart des start-ups IT ne donnent pas de SDK Dart pour le moment, alors que NodeJS est très populaire dans le monde des SDK.

Dart est cependant soutenu par Google qui

va probablement pousser de nombreuses bibliothèques dans le futur pour son langage. A suivre...

React native et JS ont donc clairement l'avantage.

À emporter / TLDR;

Ceci résume et vous aidera à choisir entre nos deux protagonistes.

Il n'y a pas de bonne ou de mauvaises réponses et cela dépendra du contexte et des préférences/connaissances de votre équipe.

Flutter et Dart

Flutter et Dart sont des choix forts pour concevoir votre application si vous avez l'intention de parier sur l'avenir.

Je ne pense pas qu'il y ait des risques à se lancer sur cette pile technologique. Les seuls risques sont qu'il vous faudra écrire/comprendre pas mal de code par vous-même, ceci parce que l'écosystème est assez récent. Vous ne pourrez pas utiliser les outils créés par les autres comme vous le feriez avec JS.

Mais Dart est un bon langage qui résout de nombreux problèmes de JavaScript.

Actuellement, je recommanderais de commencer avec Flutter pour le cross-platform si au moins certaines des phrases suivantes sont vraies pour vous :

- Vous venez d'un environnement sans JavaScript et React, et vous ne pourrez pas capitaliser sur votre expérience avec ces outils.
- Vous ne venez pas du développement web et n'avez pas l'expérience de la mise en page CSS et notamment du CSS-in-JS.
- Vous n'avez pas de délais stricts et vous avez une flexibilité dans votre livraison en termes de périmètre de fonctionnalités. Comme vous n'obtiendrez pas autant d'aide de l'extérieur que vous pourriez en obtenir avec React, il peut être plus difficile de concevoir une interface utilisateur spécifique.
- Le routage avec Flutter est également plus complexe qu'avec la fameuse librairie **react-navigation**.
- Les performances sont des caractéristiques importantes de votre application.
- Vous acceptez de perdre le bénéfice des mises à jour à-la-volée et vous reposez sur les mises à jour via le Play Store de Google ou l'Apple App Store, au moins pour les prochains mois/années.

React Native et TypeScript (ou JavaScript)

D'autre part, React Native est soutenu par Facebook et de nombreuses autres

entreprises. C'est le framework le plus avancé de sa catégorie.

Malgré l'absence de fonctionnalités telles que le typage statique en JS, vous pouvez (voir devrez) combler cela en utilisant certains outils tels que TypeScript.

Je vous recommande d'utiliser React Native si vous avez beaucoup de choses qui correspondent aux phrases suivantes :

- Vous êtes déjà un développeur **React, Vue ou Angular** et vous êtes familiers avec le concept de programmation déclarative en composants.
- Vous venez d'un environnement web et vous êtes familier avec la mise en page CSS, vous avez de l'expérience avec les bibliothèques CSS-in-JS.
- Vous aimez compter sur un grand et riche écosystème de développeurs et d'outils que vous pouvez utiliser dans votre application ; vous voulez obtenir du support externe sur des forums populaires tels que stack overflow.
- Vous souhaitez tirer parti des mises à jour à la volée à l'aide de Microsoft **Codepush** ou **Expo.io**, pour des correctifs et une livraison rapides.
- Vous voulez utiliser des outils SASS populaires qui apportent un support pour NodeJS prêt à l'emploi.

A vous de coder !

Complétez votre collection

Tarif unitaire 6,5 € (frais postaux inclus)



226



229



233



235



237



238



239

☐ 226 : ☐ ex ☐ 233 : ☐ ex ☐ 237 : ☐ ex ☐ 239 : ☐ ex
☐ 229 : ☐ ex ☐ 235 : ☐ ex ☐ 238 : ☐ ex

soit exemplaires x 6,50 € = € € soit au **TOTAL** = €

Commande à envoyer à :
Programmez! 57 rue de Gisors - 95300 Pontoise

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

E-mail : @

Règlement par chèque à l'ordre de Programmez! | Disponible sur www.programmez.com



Jean-Michel Torres
IBM Quantum France
IBM Systems Center | Montpellier

« Hello Quantum World ! »

Il est préférable de lire cet article après avoir acquis un premier niveau de connaissance du vocabulaire et du fonctionnement des qubits dans le cadre du calcul quantique. Vous pouvez vous référer au dossier paru dans le numéro 239 de Programmez!

Qu'est-ce qu'un algorithme quantique ?

« Le calcul quantique, c'est très différent, il faut tout oublier, tout reconstruire » : on commence par apprendre que la plus petite unité d'information n'est pas un bit qui peut prendre simplement deux valeurs (0 ou 1), mais un qubit (quantum bit), que celui-ci peut être dans un état de superposition des deux valeurs 0 et 1 que l'on note d'ailleurs $|0\rangle$ et $|1\rangle$ (on prononce « ket zero » et « ket un »). On nous explique ensuite que l'état d'un qubit peut être représenté sur une sphère car les coefficients de la superposition des états de base peuvent être des nombres complexes. Il s'agit de la sphère de Bloch.

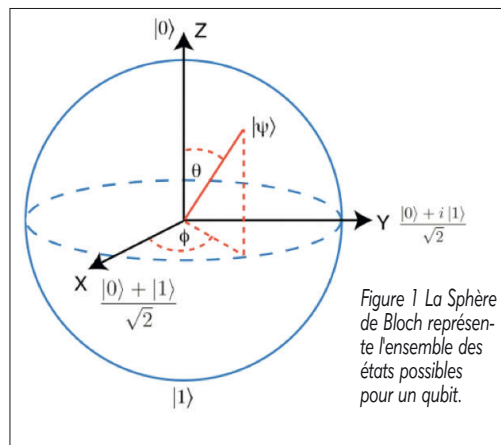


Figure 1 La Sphère de Bloch représente l'ensemble des états possibles pour un qubit.

Les « portes » entrent alors en scène permettant de manipuler les états des qubits, et même pour en intriquer plusieurs...

On connaît les portes X, Y, Z dites « de Pauli », et on fait le premier « Hello World! » avec une porte de Hadamard et une porte « CNOT », ce qui permet facilement d'observer les phénomènes de superposition et d'intrication quantiques.

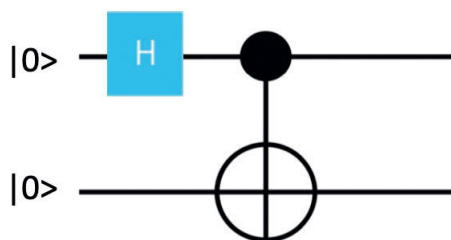


Figure 2 : Un circuit quantique réalisant la superposition quantique et l'intrication des états des deux qubits

C'est très intéressant, alors on cherche à aller un peu plus loin, on construit un circuit quantique pour réaliser un additionneur à deux qubits, éventuellement un « échange » d'état entre deux qubits à l'aide de trois CNOT tête bêche et on parle alors de « téléportation quantique » et du théorème de non-clonage. On peut aussi chercher à superposer des états de plusieurs qubits ... Mais, la question vient rapidement : que fait-on de tout ça ? Que fait-on d'utile ? Où est l'avantage quantique ?

Les algorithmes quantiques ont leurs « classiques » :

Il y a d'abord « l'algorithme de Grover » : son auteur Lov Grover a prouvé qu'il était « optimal » pour chercher si un élément donné se trouve ou pas dans une liste non ordonnée. On aborde alors la théorie de la complexité algorithmique, on évoque le problème « P est-il égal à NP ? ». Autrement dit, existe-t-il toujours un algorithme dont le temps de résolution est polynomial par rapport à la taille du problème lorsque l'on connaît un algorithme qui permet de vérifier une solution à ce même problème en un temps polynomial ?

On peut montrer que l'algorithme de Grover résout le problème pour une liste de longueur n en un temps proportionnel à \sqrt{n} plutôt qu'en un temps proportionnel à n . Et plus la liste est longue (plus n est grand) et plus l'avantage est significatif. Mais l'algorithme de Grover n'est pas très facile à comprendre, il faut passer par quelques pages de mathématiques ... On y parvient.

Il y a aussi la figure de proue de l'algorithmique quantique : « l'algorithme de Shor ». Là aussi nommé d'après son auteur : le Professeur Peter Shor du M.I.T (Massachusetts Institute of Technology). C'est le plus fameux, celui qui fait couler beaucoup d'encre depuis que son auteur a démontré en 1995 que les principes de l'algorithmique quantique permettent de trouver les diviseurs d'un nombre entier en un temps exponentiellement plus court qu'avec un algorithme « classique ». En clair : il est théoriquement capable de trouver en quelques minutes les clés privées utilisées dans les algorithmes de cryptographie les plus courants (en particulier « RSA »), là où des milliers de processeurs et des milliers (ou millions) d'années sont nécessaires actuellement. Il faudra pour cela que les machines quantiques aient atteint une capacité de calcul bien supérieure aux systèmes existant actuellement. L'algorithme de Shor est également plutôt difficile à comprendre et à étudier dans le détail.

Il y a beaucoup d'autres algorithmes quantiques, moins d'une centaine toutefois. On en trouve un recensement et une courte description sur le site « <https://quantumalgorithmzoo.org/> ».

La composante « AQUA » de la librairie de programmation open source Qiskit permettant de construire et exécuter des algorithmes quantiques dans un environnement de programmation Python permet d'en utiliser plus d'une vingtaine, comme des modules directement exécutables, sans qu'il soit nécessaire d'en réécrire les détails.

La plupart de ces algorithmes sont également assez complexes et l'effort pour les mettre en œuvre se justifie surtout dans le cadre de la R&D pour les spécialistes dans le domaine du calcul de haute performance.

A ce stade on acquiert le sentiment que les algorithmes quantiques sont un domaine encore difficile, pour les comprendre, mais également pour les créer, et les caractériser.

Ceci étant dit, je me propose d'expliquer en détail le plus simple des algorithmes quantiques, sans doute aussi un de ceux qui ont été décrits les premiers. Lui aussi porte le nom de son auteur : David Deutsch. Il permet de voir en quoi et comment le modèle de calcul quantique (avec les qubits, les propriétés de superposition et d'intrication quantiques) peut se révéler avantageux.

L'algorithme de Deutsch est en général le premier que l'on trouve expliqué dans les ouvrages consacrés au calcul quantique.

Le problème de Deutsch.

Avant de regarder l'algorithme de Deutsch, il faut définir ce qu'est « le problème » de Deutsch. Il a été considéré surtout parce qu'il présente une caractéristique intéressante pour l'algorithmique quantique : voici une question à laquelle il est possible de répondre en utilisant une seule action avec un circuit quantique, alors qu'une démarche classique aurait nécessité deux actions.

Voici donc le problème de Deutsch :

Considérant une fonction f dont l'entrée peut être soit 0 soit 1 et dont le résultat peut être soit 0 soit 1 : on ne sait rien de plus de cette fonction et on cherche à savoir si elle est constante (son résultat est toujours 0 ou bien toujours 1) ou au contraire si la fonction est équilibrée (son résultat est 0 dans un cas et 1 dans l'autre).

De manière évidente, dans le cadre de l'algorithmique classique, il faut effectuer le calcul de f pour chacune des valeurs d'entrée 0 et 1, donc à deux reprises, pour décider si la fonction f est constante ou pas.

Le problème est assez simple, son utilité réside surtout dans la possibilité de montrer un cas où l'algorithme quantique a un comportement avantageux.

Avant de passer à l'algorithme de Deutsch, constatons qu'il y a seulement quatre possibilités pour la fonction f , et nommons f_0, f_1, f_2, f_3 , les quatre fonctions correspondantes. (voir le).

x	$f_0(x)$	$f_1(x)$	$f_2(x)$	$f_3(x)$
0	0	0	1	1
1	0	1	0	1

Tableau 1 : Les quatre fonctions de $\{0,1\}$ dans $\{0,1\}$

Ici f_0 et f_3 sont constantes, f_1 et f_2 sont équilibrées.

L'algorithme de Deutsch

On a l'habitude de parler de circuit quantique, voici donc le circuit qui représente l'algorithme de Deutsch, la fonction f (dont on cherche à savoir si elle est constante ou pas) est représentée au milieu :

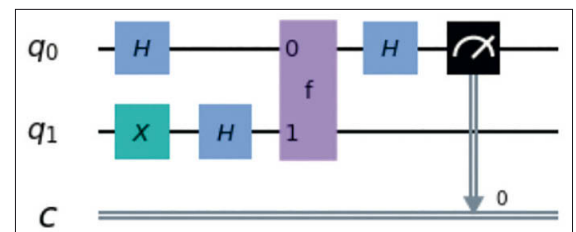


Figure 3 : Le circuit représentant l'algorithme de Deutsch pour la fonction f_i

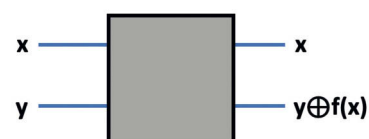
A partir de là on peut démontrer que ce circuit résout le problème de Deutsch. Il peut être intéressant de lire ce calcul, dont les formules semblent complexes surtout à cause des notations, mais qui ne dépassent pas le niveau d'un ou une élève de terminale. On peut mesurer ainsi l'importance du fondement mathématique de l'algorithmique quantique, tout autant que de l'astuce qu'a trouvée David Deutsch.

Pour ceux qui souhaitent voir ce calcul détaillé, il est disponible ici : <http://keeptheketalive.fr/plus/deutsch-calcul.pdf>

Mais ce n'est pas indispensable. On peut faire confiance au calcul et vérifier l'algorithme de Deutsch par l'expérience avec des programmes assez simples avec Python et la bibliothèque Qiskit (qiskit.org).

Retenons que pour une fonction constante, le résultat de l'algorithme sera 0, alors qu'on obtiendra le résultat 1 pour une fonction équilibrée.

Tout d'abord, il faut construire les quatre fonctions f_0, f_1, f_2 et f_3 . Nous allons les construire sous leur forme la plus simple possible (mais on pourrait imaginer des fonctions très « coûteuses » en temps de calcul, l'algorithme de Deutsch s'appliquerait de la même manière, il serait plus « utile »).

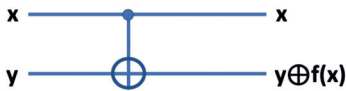


Pour f_0 : le circuit suivant convient :



En effet $y \oplus f_0(x)$ et y ont la même valeur car $f_0(x)$ vaut 0 par définition de f_0 .

Pour f_1 , le circuit suivant convient :



En effet dans ce cas $y \oplus f_1(x)$ change de valeur lorsque x change de valeur. C'est exactement la même chose pour le cas suivant avec f_2 :



Enfin, pour f_3 , ce circuit convient, car $y \oplus f_3(x)$ reste constant quand x varie :



Passons à présent à la pratique avec Python et la bibliothèque spécialisée et open source : Qiskit (quantum information science kit). L'installation de la librairie Qiskit se fait par 'pip install qiskit', mais a besoin d'autres bibliothèques pour fonctionner. C'est la raison pour laquelle le site qiskit.org recommande l'installation dans un environnement anaconda.

Pour démarrer, il faut importer ce dont on a besoin depuis la librairie qiskit (les méthodes nécessaires à la construction et à l'exécution du circuit et une cible pour l'exécution), ici on a choisi un des huit systèmes disponibles librement et gratuitement sur le Cloud d'IBM :

```
1 from qiskit import QuantumCircuit, execute, IBMQ
2 MY_API_TOKEN= ' * * coller votre token API ici * * '
3 IBMQ.save_account(MY_API_TOKEN, overwrite=True)
4 IBMQ.load_account()
5 sp = IBMQ.get_provider(hub='ibm-q')
6 backend = sp.get_backend('ibmq_vigo')
```

Les lignes 2 et 3 ci-dessus ne sont à effectuer qu'une seule fois, votre "token" est à récupérer sur le site IBM Quantum Experience (dans la rubrique Mon Compte) dès lors que vous avez créé un compte d'accès (c'est gratuit, et ça vous donne accès à des tutos, des documentations, un composeur graphique de circuits quantiques, un émulateur à 32 qbits, et 8 machines physiques).

Avec Qiskit, il est possible d'écrire les fonctions à et de les transformer en instructions réutilisables au sein de l'algorithme de Deutsch, par exemple comme ceci :

```
1 fonction0 = QuantumCircuit(2, name='f0')
2 fonction0.iden([0,1])
3 fonction0.to_instruction()
```

```
1 fonction1 = QuantumCircuit(2, name='f1')
2 fonction1.cx([0],[1])
3 fonction1.to_instruction()
```

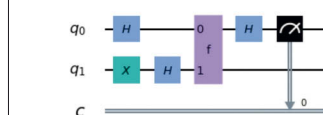
```
1 fonction2 = QuantumCircuit(2, name='f2')
2 fonction2.x([1])
3 fonction2.cx([0],[1])
4 fonction2.to_instruction()
```

```
1 fonction3 = QuantumCircuit(2, name='f3')
2 fonction3.x([1])
3 fonction3.to_instruction()
```

A présent, il est possible d'intégrer chacune de ces « instructions » (fonctions f_0, f_1, f_2, f_3) dans le circuit qui sert à effectuer l'algorithme de Deutsch.

Dans cette syntaxe, l'algorithme de Deutsch s'écrit (ici avec l'exemple de la fonction comme préalablement définie) :

```
1 Deutsch = QuantumCircuit(2,1) # circuit avec 2 qubits et
2                               # 1 bit classique pour la lecture
3 Deutsch.x([1])                # inversion du qubit 1
4 Deutsch.h([0,1])              # H sur les 2 qubits
5 Deutsch.append(fonction0,[0,1])# insertion de la fonction
6 Deutsch.h([0])                # H du qubit 0
7 Deutsch.measure([0],[0])      # lecture du qubit 0
8 Deutsch.draw(output='mpl')    # dessin du circuit, style Matplotlib
```



Le lecteur pourra vérifier tour à tour que le résultat des exécutions est bien celui attendu pour les fonctions constantes ou équilibrées. On notera que la mesure se fait sur le qubit 0, aussi surprenant que cela puisse paraître à l'observation des circuits, mais le calcul mathématique le montre clairement.

Par exemple, pour :

```
1 # define job, get results
2 job = execute(Deutsch,backend,shots=1024)
3 Deutsch_result = job.result()
4 print(Deutsch_result.get_counts(Deutsch))

{'0': 992, '1': 32}
```

Ici le résultat 0 est obtenu 992 fois, et le résultat 1 est obtenu à 32 reprises, pour un total de 1024 répétitions de l'expérience (« shots »), il s'agit de la manifestation du « bruit » dans le processeur, qui se manifeste par ce taux d'erreur (de l'ordre de 3%), c'est une autre histoire...

Voici donc un algorithme qui permet de faire en une seule fois ce qui nécessite toujours deux « passes » lorsque l'on utilise les paradigmes de la programmation dite classique.

C'est ainsi que l'on comprend bien que la programmation quantique n'est pas un simple changement de langage de programmation comme on passerait de C++ à Java. Il s'agit réellement de tout reconstruire. C'est le défi des prochaines années. Ce qui va compter, c'est que l'ensemble des technologies nécessaires progressent de manière équilibrée : le nombre de qubits, mais aussi leur qualité, la capacité à les manipuler précisément, à mesurer leur état en introduisant le moins d'erreurs possible, mais aussi l'environnement de développement et la mise au point de bibliothèques logicielles de plus en plus abstraites et utilisables dans des calculs de plus haut niveau.

La course est lancée, l'aventure s'avère passionnante !



Jean-Michel Torres
IBM Quantum France
IBM Systems Center | Montpellier

Jouer avec des qubits : superposez les initiales de votre nom avec un algorithme quantique !

Abordons quelques notions et techniques de base du calcul quantique pour les mettre à l'œuvre sous la forme d'un petit jeu, avec presque pas de mathématique ni de physique quantique.

Le jeu dans les technologies informatiques.

Lorsqu'une technologie progresse il est un signe qui ne trompe pas : les jeux fleurissent. Qu'il s'agisse d'exercices de style ou de programmation, de support de communication et de vulgarisation ou carrément d'exploitation commerciale, les exemples sont nombreux.

Mieux : l'industrie du jeu devient un moteur d'innovation (la PlayStation 3 de Sony, sortie en 2006 et vendue à 86 millions d'exemplaires embarque un processeur de type Cell à 10 cœurs qui fait figure de prouesse technologique à l'époque). Ce n'est qu'un exemple de la riche lignée des consoles de jeu. Le jeu Pong sort sur arcade en 1972, et la console HomePong est commercialisée par Atari la même année, à l'origine il s'agissait d'un exercice de programmation pour son concepteur.

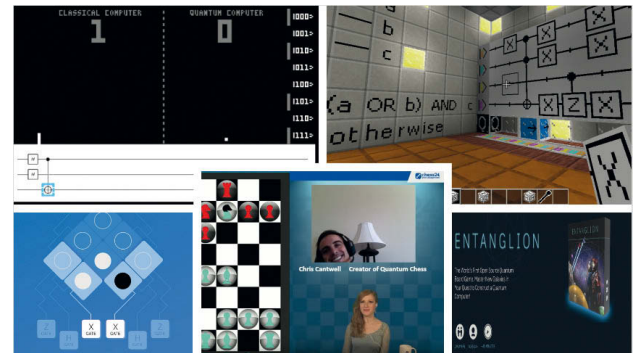
Et pourtant, le premier micro-ordinateur n'a pas encore vu le jour (d'ailleurs il est français, s'appelle Micral et sort en 1973, avant Amstrad, IBM PC et Apple II (voir « Technosaures #1 » de François Tonic !)). Plus tard, les ordinateurs et les consoles ne cesseront de rivaliser dans le domaine du jeu, chacun essayant de tirer parti de ses avantages, et forçant l'autre plateforme à innover sans cesse, les ventes se comptent en dizaines de millions d'unités.

Le calcul quantique se présente donc comme un nouvel acteur : depuis 2017 on peut jouer à « Entanglion » un jeu de plateau imaginé par des scientifiques d'IBM pour vulgariser quelques notions et un peu de vocabulaire, il s'agit de diriger son vaisseau spatial au travers d'un univers étrange où certains phénomènes quantiques se manifestent à grande échelle.

A la même époque, un jeu d'échec dont le comportement des pièces illustrent les propriétés quantiques apparaît, une partie contre une championne d'échec est à voir sur Youtube : « Anna Rudolf plays Quantum Chess ».

Sur l'Apple Store le jeu Hello Quantum permet de résoudre de petits puzzles à l'aide d'une logique inspirée des portes quantiques utilisées dans le calcul quantique, et il y a d'autres exemples : une version QPONG (disponible sur github) et on peut construire des circuits quantiques à résoudre dans Minecraft...

Je propose ici un petit défi consistant à superposer les initiales de votre nom et de votre prénom, c'est un prétexte pour apprendre quelques notions de base et programmer vos premiers circuits.



Nous allons superposer les valeurs « binaires » des code ASCII de votre prénom et de votre nom. Il s'agit donc de superposer des 1 et des 0 ou plus précisément les états quantiques de quelques qubits (quantum bits) pour former le code ASCII de deux caractères à la fois.

Le qubit est au calculateur quantique ce qu'un bit est à l'ordinateur classique : la plus petite quantité d'information traitée (en « classique », on utilise souvent les groupes de 8 bits (octets) et on les rencontre en général en très grandes quantités : GigaOctets, TeraOctets).

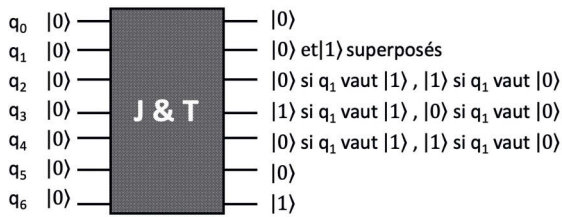
En calcul quantique, on compte encore les qubits sur les doigts de quelques mains, mais contrairement au bit qui ne peut valoir que 0 ou 1, un qubit peut se trouver dans l'état 0 ou 1, on note $|0\rangle$ et $|1\rangle$, mais aussi dans toute combinaison de ces deux états.

Pour « superposer » vos initiales, il faut d'abord connaître leur valeur ASCII (si la table des code ASCII n'est plus affichée dans votre bureau, vous la trouverez facilement sur le web). Dans cet exemple je vais utiliser « J » et « T » : dont les valeurs respectives sont 0x4A et 0x54, c'est-à-dire, en binaire : 100 1010 et 101 0100 ce sont les deux valeurs que je souhaite « superposer » :

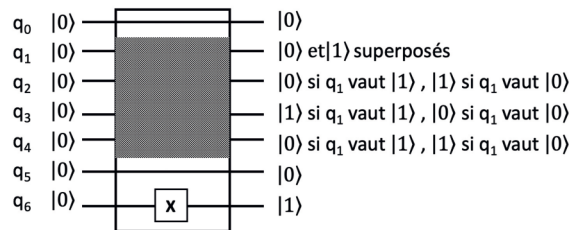
Voici le problème résumé dans ce tableau avec en vert les bits qui ont la même valeur pour le J et le T, en jaune ceux qui ont une valeur différente :

	q ₆	q ₅	q ₄	q ₃	q ₂	q ₁	q ₀
J	1	0	0	1	0	1	0
T	1	0	1	0	1	0	0

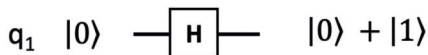
Il s'agit donc de créer un algorithme ou un circuit quantique utilisant 7 qubits (partant tous de l'état $|0\rangle$), dont les entrées et sorties peuvent être représentées de la manière suivante (pour le cas J et T) :



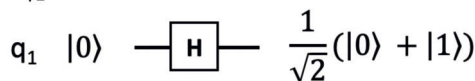
Pour le qubit 0 : l'entrée est à $|0\rangle$, je veux $|0\rangle$ en sortie, il suffit de ne rien faire
 Pour le qubit 5 (second à gauche) : c'est la même chose
 Pour le qubit 6 (le plus à gauche) : je veux toujours la valeur $|1\rangle$, comme il est à $|0\rangle$ au début, il faut lui appliquer une porte NOT (dans la panoplie des portes quantiques, elle s'appelle X).
 On a fait le plus simple et presque la moitié du chemin :



Pour les quatre autres (q_1 à q_4) : si j'obtiens une superposition de J et T, c'est que le qubit q_1 est « à moitié » dans l'état $|0\rangle$ et « à moitié » dans l'état $|1\rangle$. Il existe justement un opérateur (autrement dit une porte quantique), qui permet de réaliser ceci : la porte de Hadamard notée H (d'après le mathématicien français Jacques Hadamard). On aura donc :



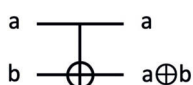
En réalité, pour des raisons de propriété des états de qubit, il y a un facteur $\frac{1}{\sqrt{2}}$, le schéma correct est :



L'idée est que l'on a un état « avec autant de 0 que de 1 dedans » et quand on va le mesurer, comme on revient dans un monde « classique » et que ceci n'a pas de sens, on va trouver comme résultat de mesure tantôt $|0\rangle$ et tantôt $|1\rangle$ sans qu'il soit possible de le prédire, et avec une probabilité $\frac{1}{2}$ pour chaque cas.

Alors, les autres qubits (q_2, q_3, q_4) doivent également être superposés mais dans un état qui dépendra de l'état de q_1 (on ne veut pas avoir q_2 à $|1\rangle$ si q_1 est à $|1\rangle$, cf. le tableau ci-dessus).

Là aussi il existe une porte quantique, sur deux qubits, qui permet d'obtenir ce résultat si s'agit de la porte Control-Not (CNOT), elle va inverser l'état d'un qubit selon l'état de l'autre :



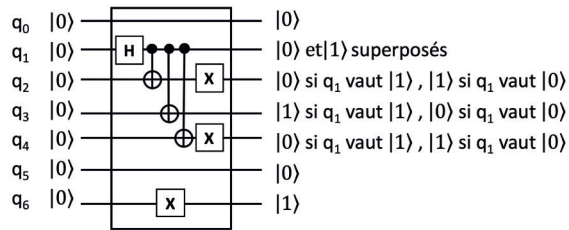
$a \oplus b$ représente le « ou exclusif » pour les valeurs de a et b.

Si je fais passer q_1 et q_3 dans une telle porte, j'aurai en sortie le ré-

sultat escompté : la sortie en face de q_3 vaudra $|1\rangle$ si q_1 vaut $|1\rangle$ et $|0\rangle$ si q_1 vaut $|0\rangle$.

Pour q_2 et q_4 c'est la même chose, sauf qu'il faudra inverser le résultat avec une porte X.

Finalement voici mon circuit (pour J et T) :



On vérifie facilement que l'on obtient soit J soit T, selon la valeur de la sortie en face de q_1 .

On peut alors utiliser la bibliothèque de programmation quantique « QISKit » pour Python, en utilisant par exemple un notebook Jupyter.

Import des objets et bibliothèques nécessaires :

```
1 from qiskit import ClassicalRegister, QuantumRegister
2 from qiskit import QuantumCircuit, execute
3 from qiskit.tools.visualization import plot_histogram
4 from qiskit import IBMQ, BasicAer
5 from qiskit.tools.jupyter import *
6 import matplotlib.pyplot as plt
7 %matplotlib inline
```

Construction du circuit :

```
1 # set up les registres et le circuit
2 qr = QuantumRegister(7)
3 cr = ClassicalRegister(7)
4 qc = QuantumCircuit(qr, cr)
5
6 # x sur les qubit qui valent 1 dans les deux cas
7 qc.x(qr[6])
8
9 # h sur le qubit de poids le plus faible qui change
10 qc.h(qr[1])
11 # cnot du qubit précédent sur tous ceux qui changent également
12 qc.cx(qr[1],qr[2])
13 qc.cx(qr[1],qr[3])
14 qc.cx(qr[1],qr[4])
15
16 # not sur ceux qui changent de manière opposée au premier
17 qc.x(qr[2])
18 qc.x(qr[4])
19
20
21 qc.barrier()
22 # mesures et dessin
23 qc.measure(qr, cr)
24 qc.draw(output='mpl')
```

Exécution :

```
1 backend = BasicAer.get_backend('qasm_simulator')
2 shots_sim = 10
3
4 job_sim = execute(qc, backend, shots=shots_sim)
5 status_sim = job_sim.result().get_counts()
6
7 def plot_letter(status, shots):
8     """code from the qiskit.org amiley demo
9     for histogram in status:
10         char = chr(int(binascii.unhexlify(status), 2)) # get string of the leftmost 7 bits and convert to an ASCII character
11         prob = status[binascii.unhexlify(status)] / shots # fraction of shots for which this result occurred
12         # create plot with all characters on top of each other with alpha given by how often it turned up in the on
13         plt.annotate(char, (0.5,0.5), va='center', ha='center', color = (0,0,0, prob), size = 300)
14         if (prob>0.05): # list prob and char for the dominant results (occurred for more than 5% of shots)
15             print(str(prob)+"%"+char)
16         plt.axis('off')
17     plt.show()
18
19 print(status_sim)
20 print(status_sim)
21 plot_letter(status_sim, shots_sim)
```



A votre tour ! Saurez-vous reconstruire le circuit et l'exécuter pour vos propres initiales ?

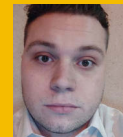
Si vous avez un nom composé ou plusieurs prénoms, c'est plus compliqué, contactez l'auteur !



Christophe Pichaud
Lead Architect
Microsoft chez Infeeny
christophep@cpixxi.com
www.windowscpp.com



Julie Lacognata
Infeeny
julie.lacognata@infeeny.com



Kevin Ansard
Infeeny
kevin.ansard@infeeny.com

Déploiement d'une application API REST sur Docker / Kubernetes sous Linux

Partie 1

Dans cet article, nous allons réaliser une application Web en .NET Core 3.1 avec un déploiement Kubernetes sous un environnement Linux. Le but de cette application est de vous montrer comment utiliser et déployer avec Kubernetes. Mais surtout de vous expliquer le concept même de Kubernetes et pourquoi ce sont, avec Docker, des technologies phares du moment.

Comme nous le disions plus haut, nous entendons de plus en plus parler de Docker et de Kubernetes. Aujourd'hui nous allons nous concentrer sur l'explication des concepts de base de Kubernetes et son utilisation sous un environnement Linux.

À l'heure actuelle, vous entendez sûrement les termes AKS, Docker, etc. Néanmoins, il faut être vigilant, car sous Azure, d'une part la visibilité sur l'utilisation de Kubernetes à un aspect « magique » qui retire aux développeurs la compréhension de l'outil, mais il y a aussi l'aspect « coût » sous Azure qui n'est pas négligeable. Ce n'est pas notre propos dans cet article. C'est pour cette raison que nous allons nous tourner vers Linux. Pourquoi ce choix ? Tout simplement, car Kubernetes dans son plus simple appareil ne fonctionne pas sous Windows. Pour les utilisateurs ayant uniquement Windows, deux choix s'offrent à vous, mais dans les deux cas il vous faut la version 19.10 d'Ubuntu que vous pouvez facilement trouver et télécharger sur le site officiel. La première solution consiste à avoir une clef USB de 8Go maximum et depuis votre PC de télécharger et d'installer Rufus, qui est un petit utilitaire vous permettant de rendre la clef USB bootable avec l'ISO dessus ce qui va vous permettre de faire un dual boot. La deuxième solution quant à elle, consiste à créer et utiliser une machine virtuelle. Nous avons choisi, ici, d'utiliser la deuxième solution en utilisant Hyper-V. 1

Une fois votre système Linux mis en place, nous allons pouvoir créer un projet en .NET Core.

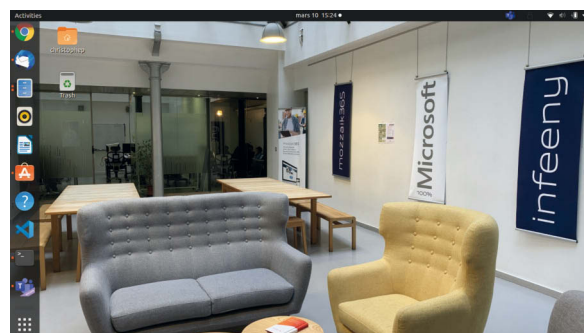
Mise en place d'un projet ASP .NET Core sous Linux

Pour réaliser ce tutoriel dans de bonnes conditions, voici la liste exhaustive des outils nécessaires :

- Visual Studio Code (IDE)
- SDK .NET Core 3.1
- Docker CE
- Helm 3.0.0
- Kubernetes via Microk8s

Ce qu'il faut savoir, c'est que sous Linux, les installations s'effectuent différemment que sous Windows.

Pour l'installation sous Ubuntu, il vous faut ouvrir votre Terminal, ainsi qu'avoir à taper ce qu'on appelle les commandes « super-droits utilisateurs » plus communément appelé **sudo** sous les OS Linux. Pour toutes installations de paquets (ou package en anglais), Ubuntu utilise la commande **apt-get install** ou **snap install**. Nous allons d'abord vous expliquer comment installer les prérequis sur votre machine virtuelle.



Installation de Visual Studio Code

VS Code est un éditeur de code Open Source développé par Microsoft. Il est compatible avec Windows, macOS et Linux. L'installation de VS Code se fait par un simple :

```
sudo snap install code --classic
```

Nous utilisons le mot clef **sudo** pour faire valoir les super-droits utilisateurs, puis nous faisons appel à la commande **snap install** pour télécharger et installer VS Code, ici nommé **code**.

Une fois que VS Code installé, nous allons ajouter l'extension C#. Pour se faire, dans VS Code cliquez sur « Extensions » puis dans la barre de recherche, tapez « C# » et installez l'extension. Nous vous conseillons fortement de faire ça pour vous simplifier le développement et l'agencement du code C#. Nous allons maintenant passer à l'installation du SDK .NET Core.

Installation du SDK .NET Core

Le .NET Core 3.1 est le framework préconisé par Microsoft pour faire les nouveaux projets NET. Nous allons donc installer ce SDK en faisant attention à bien sélectionner la version 3.1. Nous allons donc procéder de la manière suivante :

```
wget -q https://packages.microsoft.com/config/ubuntu/19.04/packages-microsoft-prod.deb -O packages-microsoft-prod.deb
```

Le mot clef **wget** va nous permettre de télécharger des paquets depuis une URL précise. Suite à l'exécution de cette commande, nous allons ensuite taper la commande :

```
sudo dpkg -i packages-microsoft-prod.deb
```

Le mot clef **dpkg** va installer les paquets.

L'installation du SDK n'est toujours pas terminée, mais presque, il nous reste quelques commandes à lancer :

```
sudo apt-get update
sudo apt-get install apt-transport-https
```

```
sudo apt-get update
sudo apt-get install dotnet-sdk-3.1
```

La deuxième ligne permet d'installer le paquet **transport-https**. Enfin, la quatrième ligne permet de télécharger et d'installer le framework .NET Core.

Installation de Docker CE

Docker est une plateforme de conteneurisation. Quand on parle de cette notion, on parle évidemment de conteneur. Les conteneurs ressemblent à des machines virtuelles ou seul le système d'exploitation ou OS aura été virtualisé et ils regroupent tout le nécessaire de notre application (peu importe qu'elle soit sous Linux ou sous Windows). En opposition aux machines virtuelles, qui elles, vont virtualiser le hardware dans son intégralité. Docker est donc une solution qualifiée de compacte, puissante, innovante et qui permet une scalabilité importante. Passons maintenant à son installation :

```
sudo apt install docker.io
```

Cette simplement comme vous permet l'installation de Docker CE (Community Edition, ndlr.). Maintenant, regardons comment démarrer une instance de Docker à l'aide des lignes suivantes :

```
sudo systemctl start docker
sudo systemctl enable docker
```

Installation de Helm

Maintenant que toutes les étapes précédentes sont bien mises en place, nous pouvons télécharger et installer Helm. Helm est un outil qui va vous permettre d'installer et de gérer le cycle de vie de vos applications Kubernetes. C'est un gestionnaire de paquets à déployer pour Kubernetes.

```
sudo snap install helm --classic
```

Installation de Kubernetes

Pour son installation sous Ubuntu, nous allons passer par Microk8s, qui est une version si l'on peut dire « light », mais aussi une version pour les développeurs de Kubernetes et qui va nous servir pour notre déploiement en local. Pourquoi une version « light » ? Tout simplement, car nos machines ne sont pas extensibles à l'infini, et c'est la même raison qui nous a poussés à vous faire installer VS Code qui est beaucoup plus léger que son camarade Visual Studio. Bref, revenons-en au sujet principal : Kubernetes ! C'est une plateforme Open Source qui a pour but de gérer la charge de travail des services conteneurisés. Autrement dit, on peut le visualiser comme un chef d'orchestre ou tout simplement comme un orchestrateur (c'est le terme le plus récurrent lorsque l'on parle de Kubernetes, NDLR.). Pour l'installer, rien de plus simple, nous allons exécuter la commande :

```
sudo snap install microk8s --classic --channel=1.17/stable
```

Par la suite, nous allons donner les droits utilisateurs sur Microk8s grâce à :

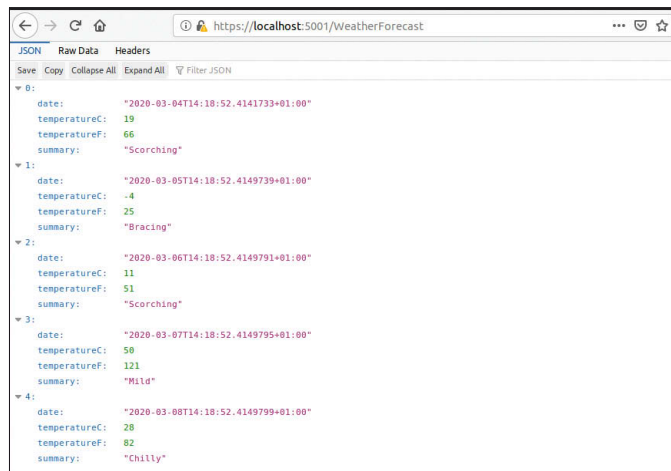
```
sudo usermod -a -G microk8s $USER
```

Enfin, pour vérifier les différents statuts de Microk8s, nous allons lancer :

```
microk8s.status --wait-ready
```

Création d'un projet REST API C#

Ouvrez votre Terminal et créer un dossier directement sur votre Bureau avec la commande suivante :



```
mkdir kubernetes_project
```

Ensuite, placez-vous dans le dossier récemment créé :

```
cd kubernetes_project
```

Maintenant, nous allons créer un projet .NET Core Web API par défaut, pour se faire, nous allons utiliser l'assistant CLI dotnet et plus précisément la commande suivante :

```
dotnet new webapi -o app
```

Une fois que votre projet est initialisé, il faut vous placer dans le dossier app :

```
cd app
```

Nous allons vérifier que notre application compile et se lance sans problème :

Petite astuce : Si vous voulez vérifier où vous vous situez en termes de path dans votre terminal, tapez **pwd**

```
dotnet run watch
```

Votre terminal va vous indiquer l'adresse et le port sur lequel il « écoute », vous devez vous rendre sur cette URL :

<https://localhost:5001/WeatherForecast/> . 2

On constate que l'API générée par défaut, par le client .NET, nous renvoie un JSON avec les informations présentes dans la solution WeatherForecast. Nous pouvons dire qu'à ce stade, nous avons une Web API .NET Core sous Linux qui fonctionne correctement. Nous allons, maintenant, modifier le code source initial, pour afficher deux variables d'environnement. Variables qui vont remplacer les températures aléatoires que nous avons juste avant.

Voici les modifications à apporter à votre classe WeatherForecast :

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Logging;

namespace app.Controllers
{
    [ApiController]
    [Route("[controller]")]

```

```

public class WeatherForecastController : ControllerBase
{
    private readonly ILogger<WeatherForecastController> _logger;

    public WeatherForecastController(ILogger<WeatherForecastController> logger)
    {
        _logger = logger;
    }

    [HttpGet]
    public InfoModel GetInfo()
    {
        return new InfoModel {
            AppEnvironment = GetEnvironmentVariable("APPENVIRONMENT"),
            AppHost = GetEnvironmentVariable("APPHOST") };
    }

    private string GetEnvironmentVariable(string name)
    {
        _logger.LogInformation($"Getting environment variable '{name}'.");
        return Environment.GetEnvironmentVariable(name.ToLower()) ??
            Environment.GetEnvironmentVariable(name.ToUpper());
    }
}

```

Et voici le Model, InfoModel :

```

using System;

namespace app.Models
{
    public class InfoModel
    {
        public string AppEnvironment {get; set;}
        public string AppHost {get; set;}
    }
}

```

Nous allons ensuite, déclarer deux variables d'environnement en copiant les deux lignes ci-dessous directement dans le terminal :

```

export APPENVIRONMENT="development"
export APPHOST="local"

```

Ces commandes ne vous afficheront rien lors de l'exécution. Par contre, si vous souhaitez vérifier qu'elles ont bien été créées, vous pouvez écrire la commande suivante :

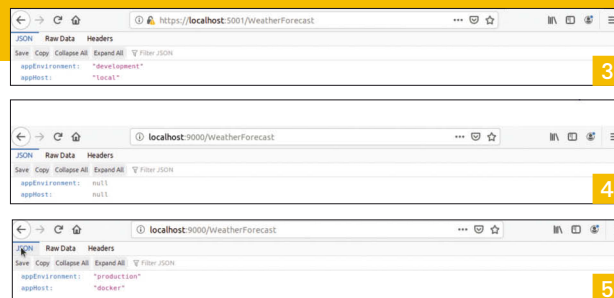
```
env | grep -e APPENVIRONMENT
```

Nous allons maintenant relancer notre application et nous pouvons constater dorénavant à l'affichage les éléments qui suivent : **3**
 Nous allons donc pouvoir mettre en place notre conteneur Docker avec notre API.

Mise en place de notre conteneur Docker avec notre API

Nous allons pour cela créer un fichier se nommant Dockerfile à la racine du dossier kubernetes_project.

Ce fichier devra ressembler à ça :



```

FROM mcr.microsoft.com/dotnet/core/sdk:3.1-alpine as build
WORKDIR /app

# copy csproj and restore
COPY app/*.csproj ./aspnetapp/
RUN cd ./aspnetapp/ && dotnet restore

# copy all files and build
COPY app/. ./aspnetapp/
WORKDIR /app/aspnetapp
RUN dotnet publish -c Release -o out

FROM mcr.microsoft.com/dotnet/core/aspnet:3.1-alpine as runtime
WORKDIR /app
COPY --from=build /app/aspnetapp/out ./
ENTRYPOINT [ "dotnet", "app.dll" ]

```

On peut observer que le Dockerfile fait appel à l'OS Alpine avec .NET Core 3.1 d'installé. Maintenant que le Dockerfile est prêt, nous allons pouvoir créer notre image. Mais tout d'abord qu'est-ce qu'une image ? Une image Docker est un fichier qui renferme un code dit exécutable ou est inclus tout le nécessaire pour le bon fonctionnement du container, les outils systèmes, les bibliothèques, les OS, etc. Dans votre terminal et dans le dossier kubernetes_project, nous allons donc commencer à utiliser l'application **docker**. Tout d'abord, nous allons créer notre image :

```
sudo docker image build --pull -t aspnet3k8s:v1
```

Ensuite nous allons créer notre conteneur et l'exécuter en lui accordant le port 9000.

```
docker run --rm -it -p 9000:80 aspnet3k8s:v1
```

Nous allons donc relancer notre application, et nous pouvons constater en nous rendant sur le localhost de toute à l'heure, que les variables d'environnement sont maintenant dans un état différent : **4**

Vous vous demandez sûrement pourquoi les variables d'environnement sont à la valeur null. On peut expliquer ça par le fait que nous nous trouvons sur notre conteneur et que celui-ci est isolé de notre machine. Pour résumer, le conteneur est une machine à part entière. Si nous voulons avoir un résultat identique de celui présent initialement, il nous faudrait redéfinir les variables d'environnement dans notre conteneur avec la commande suivante :

```
docker run --env APPENVIRONMENT=production --env APPHOST=docker --rm -it -p 9000:80 aspnet3k8s:v1
```

Cette commande va donc nous créer les variables d'environnement suivantes dans notre container et exécuter notre conteneur. Nous aurons donc ce résultat présent lorsque nous relançons l'application : **5**

La suite de cet article dans le numéro 241



Pierre-Yves Aillet
Consultant formateur
à Zenika Nantes

Découverte d'eBPF

En tant qu'utilisateur de Linux, il est possible que la seule chose qui vous intéresse soit l'inclusion de la dernière version des drivers de votre matériel dans le noyau. Pourtant, c'est loin d'être le seul pan du kernel qui évolue ! Ce sont notamment des évolutions du kernel (namespace, cgroups, ...) qui ont permis de développer les conteneurs logiciels tels que Docker. Dans cet article, je vous propose de découvrir une autre innovation majeure du kernel : eBPF (ce n'est pas moi qui le dis, mais The New Stack).

Historique

Historiquement, lorsqu'on souhaitait faire de la capture réseau sous Unix, cette opération était réalisée dans l'espace utilisateur (c'est à dire en dehors du kernel). En général, lorsqu'on souhaite capturer du trafic réseau pour identifier un problème, on ne veut pas l'ensemble du trafic, mais on souhaite filtrer selon certains critères (protocole, destination...). Pourtant en pratique, cela nécessitait de copier l'intégralité des paquets depuis le kernel vers l'espace utilisateur avant même de déterminer si un paquet était intéressant à conserver pour une session particulière de capture.

Ainsi en 1992, Steve McCanne et Van Jacobson ont publié l'article : The BSD Packet Filter: A New Architecture for User-level Packet Capture. Celui-ci introduit BPF et propose une nouvelle architecture pour la capture de paquets en espace utilisateur. Celle-ci repose sur une machine virtuelle à registres qui fonctionne dans le kernel et permet d'évaluer les règles de filtrage des paquets sans recopie des paquets. Ce mécanisme, nommé BPF, a été intégré au noyau Linux en 1997 dans la version 2.1.75. Il est disponible sur la plupart des Unix et est utilisé par des outils standards comme tcpdump pour sélectionner les paquets à capturer. À ce moment-là, la machine virtuelle BPF est assez limitée : elle ne comporte que 2 registres 32 bits, une pile de taille minimaliste et ne supporte que les sauts vers l'avant. En 2012, une évolution de bpf est proposée afin de sécuriser et filtrer certains appels systèmes en attachant à leur appel un programme bpf permettant d'évaluer le contexte et de permettre ou interdire l'appel comme cela était fait pour le traitement du trafic réseau.

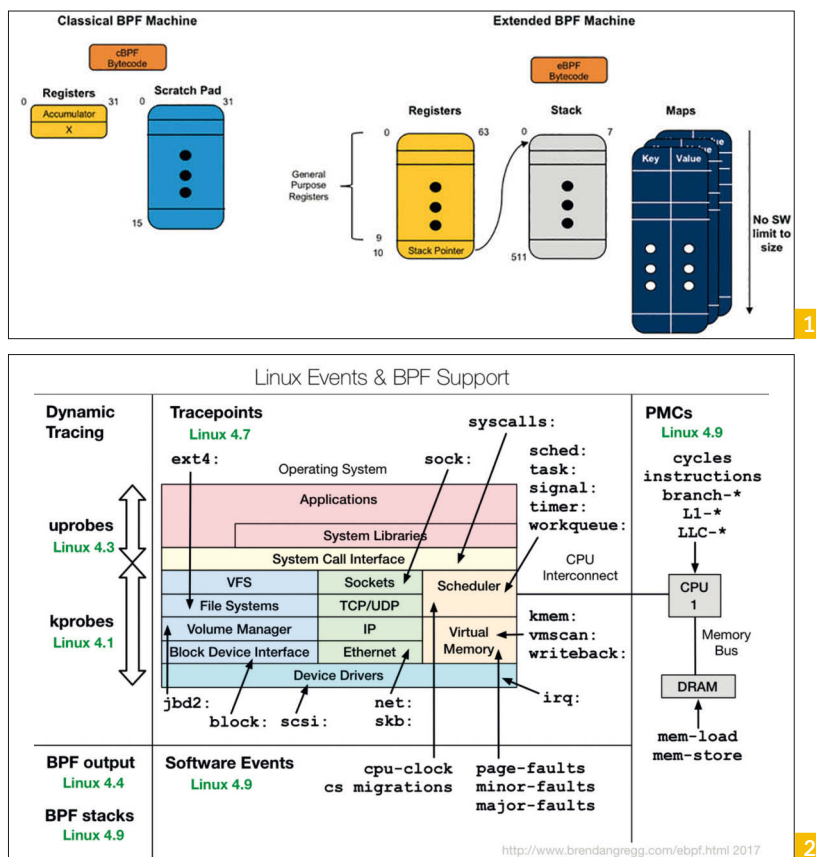
Mais l'évolution majeure arrive en 2013, Alexei Starovoitov propose alors Extended BPF. Voici un résumé de ces évolutions :

- Un vérifieur qui s'assure que le code est sûr :
 - pas de boucles infinies,
 - pas d'utilisation de pointeurs non vérifiés,
 - ...
- Un compilateur JIT pour transformer le programme BPF en code natif ;
- L'extension des capacités de la machine virtuelle :
 - passage de 2 registres 32 bits à 10 registres 64 bits,
 - possibilité d'appeler certaines fonctions du kernel. 1

Source : <https://www.netronome.com/blog/bpf-ebpf-xdp-and-bpfilter-what-are-these-things-and-what-do-they-mean-enterprise/>

Par la suite, d'autres fonctionnalités seront encore intégrées :

- utilisation de maps pour l'échange d'information entre le programme bpf et l'espace utilisateur ;
- la possibilité d'instrumenter encore plus d'événements du kernel avec des programmes bpf ;
- ...



Ces nouvelles fonctionnalités sont particulièrement intéressantes, car elles introduisent de nouveaux usages d'eBPF autres que ceux d'origines liés au réseau. Il va notamment être utilisé pour filtrer les appels systèmes accessibles aux processus avec seccomp-bpf ou ajouter du tracing sur les appels systèmes et autres événements du kernel.

Pour une liste complète, n'hésitez pas à vous référer à la page de Brendan Gregg sur le sujet. En particulier, ce schéma 2.

Pour résumer, BPF est l'appellation d'origine. Linux a introduit l'appellation eBPF et pour différencier cette version de l'ancienne parfois nommée cBPF. Donc :

- lorsqu'on vous parle de cBPF, c'est sans les extensions ;
- lorsqu'on vous parle d'eBPF, il s'agit de la version moderne ;
- lorsqu'on vous parle de BPF `_(_)/` .

Principe

Maintenant qu'on en connaît un peu plus sur l'historique du projet, voyons comment il fonctionne...

Fonctionnement

Tout est basé sur un appel système :


```
1#include
2
3int bpf(int cmd, union bpf_attr *attr, unsigned int size);
```

Le premier argument `cmd` indique l'action à réaliser, exemple : `BPF_PROG_LOAD`, pour charger un programme bpf. Le deuxième argument `attr` porte les paramètres de l'action à réaliser, sa structure dépend de la commande (valeur du premier argument). Le dernier paramètre `size` est la taille de la structure passée en deuxième argument.

Les programmes eBPF sont événementiels, c'est-à-dire que leur exécution est déclenchée en réponse à des actions ou appels de fonctions internes du kernel. Afin de conserver des données entre les différentes exécutions du programme, mais aussi afin d'échanger des informations entre le programme BPF qui tourne dans l'espace du noyau et l'espace utilisateur, eBPF propose d'utiliser des maps. Les autres commandes utilisables avec BPF sont dédiées à la création et à la manipulation de ces maps.

Mais avant d'aller plus loin, voyons comment ils sont chargés les programmes eBPF et comment ils sont reliés aux événements qui nous intéressent. Pour commencer, voyons le détail de la structure qui porte les paramètres de l'appel à `BPF_PROG_LOAD` :

```
1 struct { /* Used by BPF_PROG_LOAD */
2     __u32    prog_type;
3     __u32    insn_cnt;
4     __aligned_u64 insns; /* 'const struct bpf_insn **' */
5     __aligned_u64 license; /* 'const char **' */
6     __u32    log_level; /* verbosity level of verifier */
7     __u32    log_size; /* size of user buffer */
8     __aligned_u64 log_buf; /* user supplied 'char **'
9                          buffer */
10    [...]
11};
```

- `prog_type`, permet d'indiquer le type de programme et par quel type d'évènement il sera déclenché. Par exemple, pour attacher un programme à l'exécution d'une fonction on utilisera : `BPF_PROG_TYPE_KPROBE`.
- `insn_cnt` indique le nombre d'instructions du programme.
- `insns` pointe vers la liste des instructions.
- `license` indique la license du programme.
- les 3 attributs suivants (`log_level`, `log_size` et `log_buf`) permettent d'obtenir des informations sur le chargement du programme et le résultat du vérifieur.

À moins, que vous souhaitiez directement écrire le bytecode de votre programme, il sera préférable d'utiliser LLVM afin de transformer le code C en bytecode bpf. Vous utiliserez alors une commande du type :

```
1 clang -O2 -emit-llvm -c bpf.c -o - | llc -march=bpf -filetype=obj -o bpf.o
```

Une fois la structure correctement alimentée et l'appel système effectué, le noyau prendra en charge votre programme qui subira encore quelques manipulations/transmutations :

- dans un premier temps, le vérifieur va s'assurer que le programme :
 - ne comporte pas plus d'instructions que la limite (4096 en Linux 4.14),
 - est un Diagramme orienté acyclique. C'est à dire, qu'il ne comporte pas de boucles,

- accède uniquement à des zones mémoires identifiées,
- ...

ensuite, avant sa première exécution, le programme sera transformé de bytecode eBPF vers le code natif de la plate-forme pour les architectures supportées.

BCC : BPF Compiler Collection

Tout cela peut sembler un peu compliqué à mettre en oeuvre. Heureusement, il existe un outil qui simplifie grandement l'utilisation d'eBPF : BPF Compiler Collection. Ce projet propose notamment un frontend Python que nous allons utiliser pour mettre en oeuvre quelques exemples de programmes eBPF.

Pour commencer, assurez-vous d'avoir une machine Linux avec un kernel récent (≥ 4.15). Vous pouvez également utiliser vagrant avec le Vagrantfile disponible sur le repository github contenant les exemples de cet article.

Ensuite, installez les dépendances nécessaires (cette étape n'est pas nécessaire si vous utilisez vagrant) :

```
1 $ apt update
2 $ apt install -y python python-pip bpffcc-tools
```

Pour tester avec un cas simple, créez le fichier `first_trace.py` ci-après. Ce script Python va créer et charger un programme eBPF à partir du code source passé en paramètre. Ce programme sera attaché au syscall `clone` qui est utilisé dès que l'on souhaite créer un nouveau processus.

```
1 #!/usr/bin/env python
2
3 import os
4 from bcc import BPF
5
6 print('Launching in background, pid: ', os.getpid())
7
8 # This may not work for 4.17 on x64, you need replace kprobe__sys_clone with kprobe
   __x64_sys_clone
9 BPF(text="""
10 int kprobe__sys_clone(void *ctx) {
11     bpf_trace_printk("Hello, eBPF!\n");
12     return 0;
13 }
14 """).trace_print()
```

Lancez ce script en arrière-plan avec la commande : `sudo ./first_trace.py &`

Il est nécessaire de le lancer en tant que `root` afin de pouvoir utiliser l'appel système `bpf`.

Vous pouvez éventuellement déjà voir des messages apparaître dès qu'un nouveau processus est créé. Vous pouvez également lancer le script ci-dessous qui créera un nouveau processus toutes les 3 secondes. Ce nouveau processus affiche un message et s'arrête.

```
1 #!/usr/bin/env python
2
3 import os
4 import time
5
6 def child():
```

```

7 print('New child ', os.getpid())
8 os._exit(0)
9
10 def parent():
11     while True:
12         newpid = os.fork()
13         if newpid == 0:
14             child()
15         else:
16             time.sleep(3)
17             os.waitpid(newpid, 0)
18
19
20 parent()

```

Voici un exemple de sortie de ce test :

```

1 vagrant@ubuntu-bionic:~$ sudo ./first_trace.py &
2 [1] 8178
3 vagrant@ubuntu-bionic:~$
4 ('Launching in background, pid: ', 8179)
5
6 vagrant@ubuntu-bionic:~$ ./clone.py
7 bash-8154 [000] .... 207.972080: 0x00000001: Hello, eBPF!
8 -8180 [000] .... 207.987895: 0x00000001: Hello, eBPF!
9 ('New child ', 8181)
10 python-8180 [000] .... 210.990015: 0x00000001: Hello, eBPF!
11 ('New child ', 8182)
12 python-8180 [000] .... 213.992402: 0x00000001: Hello, eBPF!
13 ('New child ', 8183)

```

Une fois votre test effectué, lancez `sudo kill <pid>`, où vous remplacez `pid` par l'identifiant du processus lancé en arrière-plan qui s'est affiché après son lancement.

Cas d'utilisation

La technologie eBPF est en plein essor, le fait de pouvoir exécuter du code en mode kernel intéresse beaucoup, d'autant plus qu'avec eBPF, il n'est pas nécessaire de recompiler le noyau ou d'être spécialiste du développement de modules pour pouvoir le faire. Ainsi, le projet est utilisé pour :

- De la capture d'événements du kernel, pour :
 - des mesures de performance,
 - du tracing.
- Du filtrage réseau :
 - haute-performance (anti-DDOS, ...),
 - avancé et dépendant du contexte.

Conclusion

J'espère que ce petit tour d'horizon vous a donné envie d'aller plus loin. Si c'est le cas, n'hésitez pas à consulter les références que j'ai consultées pour le rédiger.

Références

Historique :

Papier original par Steven McCanne et Van Jacobson :

<http://www.tcpdump.org/papers/bpf-usenix93.pdf> et <https://www.usenix.org/legacy/publications/library/proceedings/sd93/mccanne.pdf>

BPF – in-kernel virtual machine : http://vger.kernel.org/netconf2015Starovoirov-bpf_collabsummit_2015feb20.pdf

Articles :

What can BPF do for you ? :

<https://events.static.linuxfound.org/sites/events/files/slides/iovisor-lc-bof-2016.pdf>

Awesome eBPF : <https://github.com/zoidbergwill/awesome-ebpf>

How I ended up writing eBPF : <https://bolinfest.github.io/opensnoop-native>

eBPF, past, present and future : https://ferrisellis.com/content/ebpf_past_present_future/

eBPF, syscalls and map types :

https://ferrisellis.com/content/ebpf_syscall_and_maps/

Actualité :

<https://thenewstack.io/linux-technology-for-the-new-year-ebpf/>

Documentation :

https://en.wikipedia.org/wiki/Berkeley_Packet_Filter

Documentation officielle BPF :

<https://www.kernel.org/doc/html/latest/bpf/index.html>

BPF and XDP Reference Guide : <https://cilium.readthedocs.io/en/latest/bpf/>

Vérifier : <https://blogs.oracle.com/linux/notes-on-bpf-5>

Tracing :

<http://www.linuxembedded.fr/2019/03/les-secrets-du-traceur-ebpf/>

Java Flame graphs : <https://www.youtube.com/watch?v=saCGp-T6saQ>

Jérémie Lagarde DevxxxFr : <https://www.youtube.com/watch?v=rdmHrQGQIw>

Liz Rice : <https://www.youtube.com/watch?v=4SiWL5tULnQ>

Network :

XDP : [Netronome — https://github.com/Netronome/bpf-samples](https://github.com/Netronome/bpf-samples)

Cloudflare : <https://blog.cloudflare.com/ebpf-sockets-hop-distance/>

Facebook : http://vger.kernel.org/lpc_net2018_talks/ebpf-firewall-LPC.pdf

Cilium : <https://cilium.io/>

Calico : <https://www.projectcalico.org/tigera-adds-ebpf-support-to-calico/>

Tcpdump : <https://medium.com/@cjoudrey/capturing-http-packets-the-hard-way-b9c799bfb6>

Toward a faster Iptables in eBPF :

<https://webthesis.biblio.polito.it/8475/1/tesi.pdf>

Toward an eBPF-based clone of iptables :

Slides : <https://www.astrid-project.eu/images/Toward%20an%20eBPF-based%20clone%20of%20iptables.pdf>

Article complet : https://sebymiano.github.io/documents/21-Securing_Linux_with_a_Faster_and_Scalable_Iptables.pdf

Les liens de l'article :

<https://thenewstack.io/linux-technology-for-the-new-year-ebpf/>

<http://www.tcpdump.org/papers/bpf-usenix93.pdf>

<https://lwn.net/1998/0212/>

<https://www.tcpdump.org/>

<https://lwn.net/Articles/475043/>

<https://lwn.net/Articles/575531/>

https://prototype-kernel.readthedocs.io/en/latest/bpf/ebpf_maps.html

https://www.kernel.org/doc/html/v4.16/userspace-api/seccomp_filter.html

<http://www.brendangregg.com/ebpf.html>

<https://lvm.org/>

https://fr.wikipedia.org/wiki/Graphe_orient%C3%A9_acyclique

<https://www.kernel.org/doc/Documentation/features/core/eBPF-JIT/arch-support.txt>

<https://github.com/iovisor/bcc>

<https://www.vagrantup.com/>

<https://github.com/pyaillet/ebpf-discovery/blob/master/Vagrantfile>

<https://github.com/pyaillet/ebpf-discovery>



Laurent Ellerbach
Principal Engineer Manager
Microsoft
laurelle@microsoft.com
<https://github.com/Ellebach>

.NET Core IoT : du .NET Core multiplateforme avec des centaines de capteurs et périphériques déjà codés

.NET Core est un framework multiplateforme qui fonctionne aussi bien sur Windows, macOS que Linux. Il est disponible sur de nombreuses architectures processeurs allant des x86 aux ARMv7+. .NET Core est selon la dernière étude de Stack Overflow la plateforme la plus aimée des développeurs(*). Elle est également open source, maintenue par Microsoft et sous licence MIT : <https://github.com/dotnet/core>. .NET Core est dans sa version 3.1 avec des mises à jour régulières et une communauté très active. D'ici fin 2020, l'ensemble des frameworks .NET ne feront plus qu'un et sur la base de .NET Core.

(*) <https://insights.stackoverflow.com/survey/2019/#technology--most-loved-dreaded-and-wanted-other-frameworks-libraries-and-tools>

Aujourd'hui vient s'ajouter à la plateforme .NET Core IoT. Comme son nom l'indique il permet de faire de l'IoT avec .NET Core. Une préversion est sortie il y a un an et la communauté a rapidement grossi et le nombre de capteurs et périphériques ne cesse de croître. Là aussi, tout est open source, sous licence MIT : <https://github.com/dotnet/iot>.

.NET Core IoT fournit des accès natifs aux GPIO (General Purpose Input/Output), I2C (Inter-Integrated Circuit), PWM (Pulse Width Modulation), SPI (Serial Peripheral Interface) pour les plateformes Windows et Linux. Cela implique de pouvoir développer en C# des accès natifs à tous ces protocoles pour de l'embarqué industriel ou juste pour son Raspberry Pi à la maison.

Un des principaux intérêts est d'avoir des implémentations de qualités, homogènes de différents capteurs et périphériques. On y retrouve les traditionnels capteurs de températures tels les DHT, des matrices Led RGB, lecteur RFID, support pour les servo-moteurs, mais aussi des shields tels que GrovePi, BrickPi3, SenseHat ou GoPiGo3 ! La liste est disponible ici :

<https://github.com/dotnet/iot/tree/master/src/devices>

Je suis personnellement impliqué depuis le début de l'aventure et second contributeur ayant ajouté le plus de lignes de codes et l'implémentation de nombreux capteurs. La dernière implémentation est le support du FT4222(1) de FTDI et permet un support de GPIO, SPI et I2C nativement sur un Windows ou Linux « normal ». J'ai acheté ma board chez BitWizard(2). En gros, ce dongle fonctionne sur USB et offre donc le support de ces différents protocoles. Cela ouvre donc des perspectives.

Avec la puissance des machines et même si le port USB peut limiter, dans la pratique, cela fonctionne parfaitement bien sur tout ce que j'ai eu l'occasion de tester ! Et pour la mise au point de votre code ou le développement de nouveaux capteurs, ce n'est que du bonheur :-). C'est quand même beaucoup plus simple de debugger sous un vrai Windows 64 bit qu'en SSH sur le Raspberry Pi ! Ceci dit, attention, le debug bas niveau dans le développement bas niveau n'est pas facile, il est difficile de mettre un point d'arrêt au

milieu d'un protocole de communication fonctionnant avec des timings serrés.

Un peu de code

Tous les capteurs, shields et autres éléments viennent tous avec des exemples d'utilisation. Ils contiennent un schéma avec les composants nécessaires. La documentation permet donc très rapidement de démarrer un PoC.

Prenons l'exemple d'un capteur de température DHT11, grand classique dans le monde de l'IoT. Voici ce qu'il faut faire pour lire la température et l'humidité :

```
// GPIO Pin
using (Dht11 dht = new Dht11(26))
{
    Temperature temperature = dht.Temperature;
    double humidity = dht.Humidity;
}
```

Rien de plus ! Toute la complexité du protocole Wire est complètement caché. Le framework propose même une classe Température facilitant les conversions entre les différentes unités.

Autre exemple utilisant un BNO055 qui est une centrale inertielle communiquant en utilisant le protocole I2C :

```
I2cDevice i2cDevice = I2cDevice.Create(new I2cConnectionSettings(1, Bno055Sensor.DefaultI2cAddress));
Bno055Sensor bno055Sensor = new Bno055Sensor(i2cDevice);
Console.WriteLine($"Id: {bno055Sensor.Info.ChipId}, AccId: {bno055Sensor.Info.AcceleratorId}, GyroId: {bno055Sensor.Info.GyroscopeId}, MagId: {bno055Sensor.Info.MagnetometerId}");
Console.WriteLine($"Firmware version: {bno055Sensor.Info.FirmwareVersion}, Bootloader: {bno055Sensor.Info.BootloaderVersion}");
Console.WriteLine($"Temperature source: {bno055Sensor.TemperatureSource}, Operation mode: {bno055Sensor.OperationMode}, Units: {bno055Sensor.Units}");
Console.WriteLine($"Powermode: {bno055Sensor.PowerMode}");
```

Le code ci-dessus permet d'initialiser le capteur et d'afficher quelques-unes de ces propriétés. Quelques lignes sont nécessaires

(1) <https://www.ftdichip.com/Products/ICs/FT4222H.html>

(2) <http://bitwizard.nl/shop/FT4222h-Breakout-Board?search=ft4222>

en plus pour le calibrer et ensuite, il est possible d'accéder très simplement à l'ensemble des 9 axes de ce capteur :

```
while(!Console.KeyAvailable)
{
    Console.Clear();
    var magneto = bno055Sensor.Magnetometer;
    Console.WriteLine($"Magnetomer X: {magneto.X} Y: {magneto.Y} Z: {magneto.Z}");
    var gyro = bno055Sensor.Gyroscope;
    Console.WriteLine($"Gyroscope X: {gyro.X} Y: {gyro.Y} Z: {gyro.Z}");
    var accele = bno055Sensor.Accelerometer;
    Console.WriteLine($"Acceleration X: {accele.X} Y: {accele.Y} Z: {accele.Z}");
    var orien = bno055Sensor.Orientation;
    Console.WriteLine($"Orientation Heading: {orien.X} Roll: {orien.Y} Pitch: {orien.Z}");
    var line = bno055Sensor.LinearAcceleration;
    Console.WriteLine($"Linear acceleration X: {line.X} Y: {line.Y} Z: {line.Z}");
    var gravity = bno055Sensor.Gravity;
    Console.WriteLine($"Gravity X: {gravity.X} Y: {gravity.Y} Z: {gravity.Z}");
    var qua = bno055Sensor.Quaternion;
    Console.WriteLine($"Quaternion X: {qua.X} Y: {qua.Y} Z: {qua.Z} W: {qua.W}");
    var temp = bno055Sensor.Temperature.Celsius;
    Console.WriteLine($"Temperature: {temp} °C");
    Thread.Sleep(100);
}
```

Autre exemple : un capteur ADXL345 qui est un accéléromètre 3 axes. Il utilise le protocole SPI :

```
SpiConnectionSettings settings = new SpiConnectionSettings(0, 0)
{
    ClockFrequency = Adxl345.SpiClockFrequency,
    Mode = Adxl345.SpiMode
};
var device = SpiDevice.Create(settings);
// set gravity measurement range ±4G
using (Adxl345 sensor = new Adxl345(device, GravityRange.Range04))
{
    // read acceleration
    Vector3 data = sensor.Acceleration;
    //use sensor
}
```

Vous l'aurez compris, la création d'un capteur utilisant un protocole que ce soit GPIO, I2C ou SPI est vraiment très simple. Et les capteurs ont des propriétés homogènes quand ils sont de classes similaires. Un gros travail de fond a été fait pour que cela soit le plus simple possible à utiliser.

Debugger sur Windows/Linux 64 bits

Grâce à l'implémentation du FT4222, il est possible de facilement debugger sous une version 64 ou 32 bits de Windows ou Linux. Il faut d'abord installer une librairie de FTDI en suivant les instructions du repository. Et ensuite, c'est aussi simple que pour les exemples précédents.

Reprenons l'exemple du BNO055. Pour l'initialiser sur un Win-

dows/Linux 32/64 bits, il suffit de créer le driver I2C et de le passer au BNO055 :

```
var winFtdil2c = new Ft4222I2c(new I2cConnectionSettings(0, Bno055Sensor.DefaultI2cAddress));

Bno055Sensor bno055Sensor = new Bno055Sensor(winFtdil2c);
Console.WriteLine($"Id: {bno055Sensor.Info.ChipId}, AcclId: {bno055Sensor.Info.AcceleratorId}, GyroId: {bno055Sensor.Info.GyroscopeId}, MagId: {bno055Sensor.Info.MagnetometerId}");
Console.WriteLine($"Firmware version: {bno055Sensor.Info.FirmwareVersion}, Bootloader: {bno055Sensor.Info.BootloaderVersion}");
Console.WriteLine($"Temperature source: {bno055Sensor.TemperatureSource}, Operation mode: {bno055Sensor.OperationMode}, Units: {bno055Sensor.Units}");
Console.WriteLine($"Powermode: {bno055Sensor.PowerMode}");
```

La différence est donc minime. Il en va de même pour les périphériques SPI. Et pour GPIO, si le périphérique ne supporte pas un créateur permettant de spécifier les driver, il est nécessaire de faire cette modification dans le code source.

Évidemment, cela permet aussi de transformer un vrai PC sous Windows en véritable PC embarqué accédant à divers capteurs. On peut très bien imaginer faire tourner une application en ASP.NET Core et accéder à un capteur de température et l'humidité le tout tournant sur un Windows 10 ! Il en va de même pour les versions serveurs. Remplacez Windows par Linux et vous avez exactement les mêmes possibilités.

Debugger depuis Visual Studio sous Windows un Raspberry Pi sous Linux

Un cas assez courant : quand on met au point du code IoT c'est le debug sur le périphérique cible. Je vais prendre l'exemple d'un Pi sous Linux, mais c'est exactement la même chose pour n'importe quel périphérique embarqué sous Linux.

.NET Core offre la possibilité de packager tout ce qui est nécessaire dans un répertoire et ne nécessite pas son installation sur le périphérique cible. Cela permet un déploiement en simple copier/coller. Pour cela, il faut compiler le projet avec la directive *publish* et la target *-r linux-arm*

```
dotnet publish -r linux-arm /p:ShowLinkerSizeComparison=true -o .\publish
```

/p:ShowLinkerSizeComparison=true permet d'optimiser la taille. Le tout sera mis dans la racine du projet dans un répertoire *publish* car spécifié avec la directive *-o*

Vous pouvez copier le répertoire créé avec tous les fichiers sur le Raspberry Pi avec la méthode de votre choix. En utilisant WinSCP par exemple ou alors avec pscp. Ce qui donnerait quelque chose comme cela :

```
pushd .\publish
pscp -pw raspberry -v -r .\* pi@192.168.1.122:/home/pi/demonetcore
popd
```

« pushd » permet de mettre en haut de la pile un répertoire actif, tout ce qui se passe dans les lignes de commande suivantes le sont

depuis ce répertoire jusqu'à rencontrer le « popd » qui va l'enlever. Le précédent répertoire actif sera alors utilisé. C'est très pratique pour ce genre de manipulations. Côté code, tout est prêt. Il faut maintenant installer les debug tools sur le Pi :

```
curl -sSL https://aka.ms/getvsdbgsh | /bin/sh /dev/stdin -v latest -l ~/vsdbg
```

Cela permet d'installer les outils de debug de Visual Studio qui vont permettre le debug local, mais aussi à distance. À noter ce que outils peuvent être utilisés pour debugger à peu près tout et n'importe quoi. Une fois installé, il va falloir créer un mot de passe pour le root :

```
sudo passwd root
```

Puis permettre au root de se connecter. Pour cela, il faut éditer le fichier suivant :

```
sudo nano /etc/ssh/sshd_config
```

Et ajouter la ligne suivante:

```
PermitRootLogin yes
```

On reboote :

```
sudo reboot
```

Il faut ensuite lancer le programme à debugger sur le Raspberry puis s'y connecter avec Visual Studio : **1**

Il faut bien faire attention de sélectionner « Attach to Process » puis de choisir « SSH » et de rentrer « root » comme nom d'utilisateur avec le mot de passe choisi précédemment. La liste des processus de l'utilisateur « root » s'affiche, il faut penser à cocher la case « Show process from all users » pour aller chercher l'application que l'on veut debugger. Et cliquer sur « Attach ». **2**

Ensuite, le debug se passe comme dans la vraie vie, c'est possible de mettre des points d'arrêts, d'avoir les valeurs des variables. **3** Attention : récupérer des valeurs prend un peu de temps. Le Pi n'est pas une machine très puissante. Il permet tout de même de nombreux scénarios. Il m'a été utile dans la mise au point des nombreux shields et capteurs que j'ai écrits.

Et les containers ?

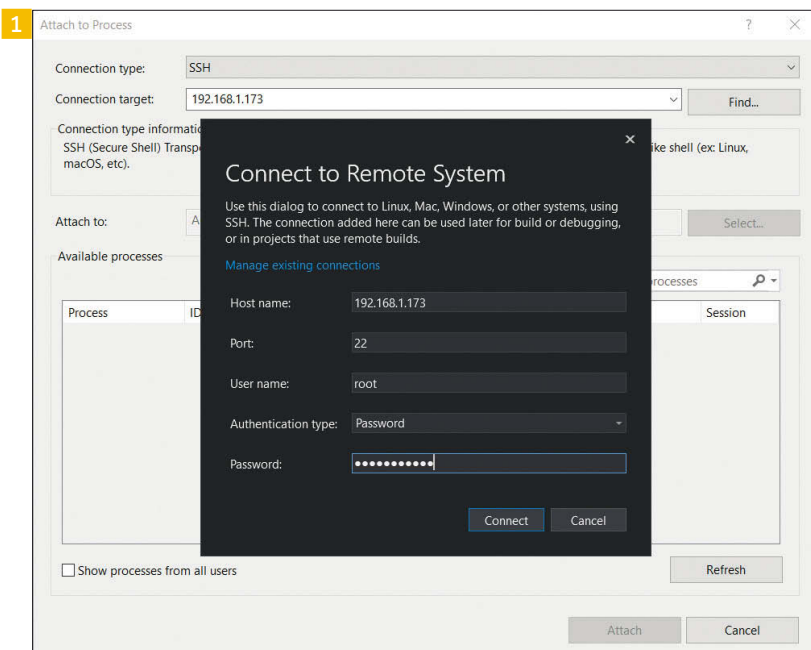
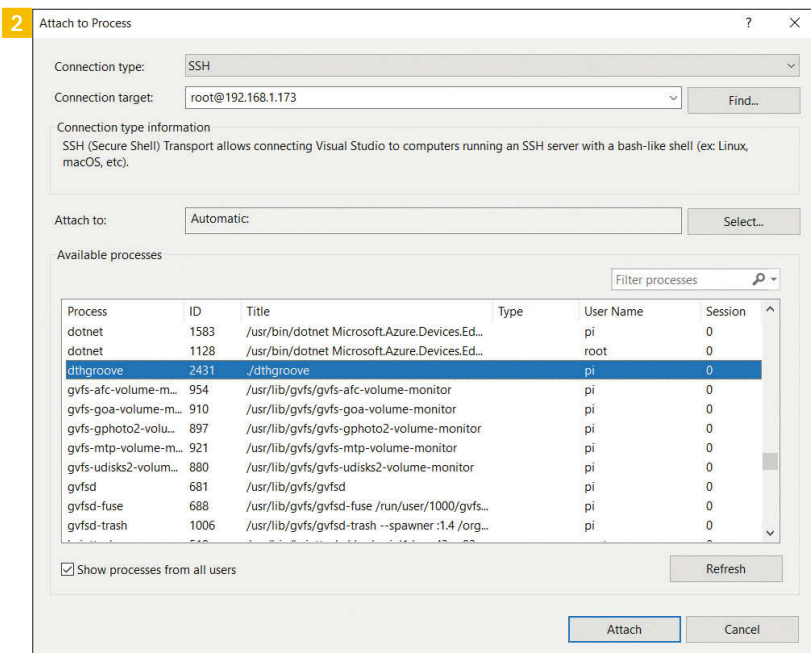
.NET Core IoT fonctionne parfaitement en container. Je les utilise tous les jours (ok, surtout en été :-)) avec mon arrosage automatique, cf. <https://github.com/Ellebach/Sprinkler-IoTEdge>. Le principe est d'utiliser Azure IoT Edge pour le déploiement sur un Pi. Ayant plusieurs maisons et appartements de mes amis utilisant mon système, cela me permet de déployer une seule mise à jour dans mon Azure IoT Edge et d'avoir un déploiement automatique sur l'ensemble des périphériques. Très pratique.

Et pour en revenir à .NET Core IoT, j'utilise dans le code de l'arrosage automatique, les ports GPIO pour ouvrir et fermer des électrovannes qui ouvrent des électrovannes.

J'utilise le même principe que décrit plus haut. Le code complet se trouve ici : <https://github.com/Ellebach/Sprinkler-IoTEdge/blob/master/Sprinkler-NetCore/Models/Sprinkler.cs>

Pour faire simple, j'ouvre donc un des GPIO (_sprinkler), je vérifie si mon électroaimant est inversé ou pas, c'est-à-dire si son état fermé est avec ou sans courant et puis j'écris la valeur high ou low pour faire changer d'état l'électroaimant :

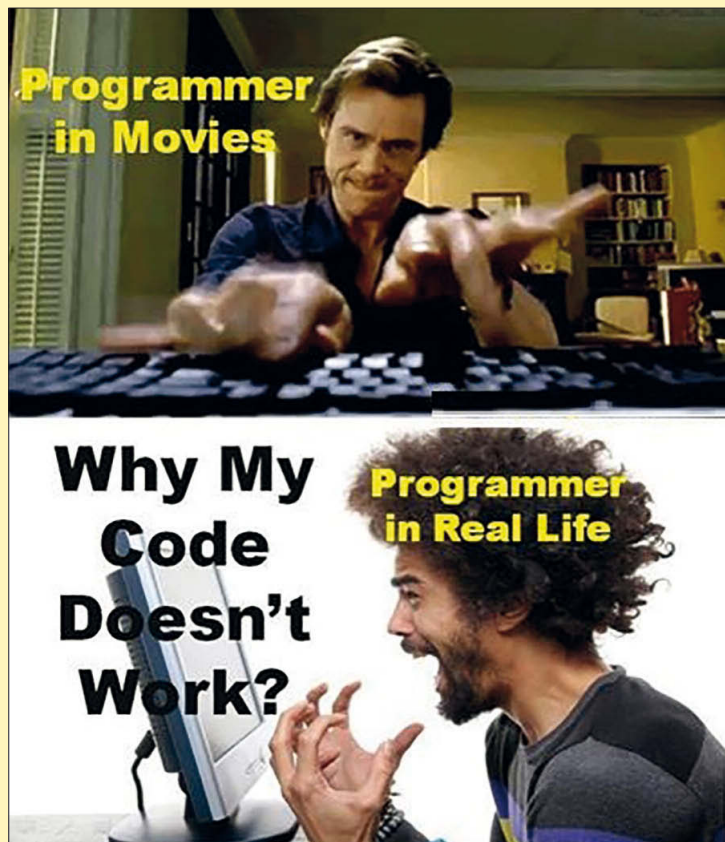
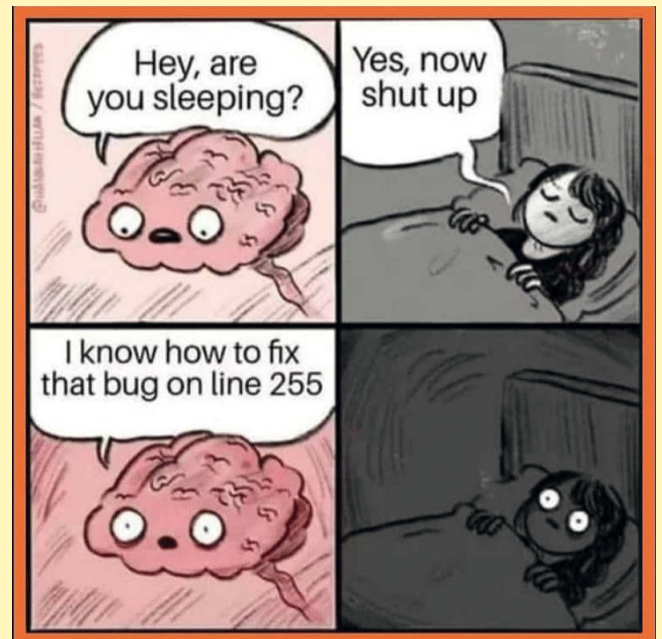
```
_controller = new GpioController();
_controller.OpenPin(_sprinkler, PinMode.Output);
_pinValue = IsInverted ? PinValue.High : PinValue.Low;
_controller.Write(_sprinkler, _pinValue);
```



```

DemoNetCore.csproj Program.cs
DhtTx
33 private readonly DhtType _dhtType;
34 private I2cDevice _sensor;
35 private GpioController _controller;
36
37 private Stopwatch _stopwatch = new Stopwatch();
38
39 private int _lastMeasurement = 0;
40
41 /// <summary>
42 /// How last read went, <c>true</c> for success, <c>false</c> for failure
43 /// </summary>
44 18 references | Zhang Yuexin, 20 days ago | 1 author, 2 changes | 2 work items
45 public bool IsLastReadSuccessful { get; private set; }
46
47 /// <summary>
48 /// Get the last read temperature
49 /// </summary>
50 /// <remarks>
51 /// If last read was not successful, it returns double.NaN
52 /// </remarks>
53 1 reference | Zhang Yuexin, 71 days ago | 1 author, 1 change | 1 work item
54 public Temperature Temperature
55 {
56     get
57     {
58         ReadData();
59
60         switch (_dhtType)
61         {
62             case DhtType.Dht11:
63                 return Temperature.FromCelsius(GetTempDht11());
64             case DhtType.Dht12:
65             case DhtType.Dht21:
66             case DhtType.Dht22:
67                 return Temperature.FromCelsius(GetTempDht22());
68             default:
69                 return Temperature.FromCelsius(double.NaN);
70         }
71     }
72 }
73
74 /// <summary>
75 /// Get the last read of relative humidity in percentage
76 /// </summary>
77 /// <remarks>
78 /// If last read was not successful, it returns double.NaN
79 /// </remarks>
80 1 reference | Zhang Yuexin, 71 days ago | 1 author, 1 change | 1 work item
81 public double Humidity
82 {
83     get
84     {
85         ReadData();
86
87         switch (_dhtType)
88         {
89             case DhtType.Dht11:
90                 return Humidity.FromPercentage(GetHumDht11());
91             case DhtType.Dht12:
92             case DhtType.Dht21:
93             case DhtType.Dht22:
94                 return Humidity.FromPercentage(GetHumDht22());
95             default:
96                 return Humidity.FromPercentage(double.NaN);
97         }
98     }
99 }
100
101 private void ReadData()
102 {
103     _stopwatch.Restart();
104     try
105     {
106         _sensor.ReadData();
107     }
108     catch { }
109     _lastMeasurement++;
110     IsLastReadSuccessful = true;
111 }
112
113 private double GetTempDht11()
114 {
115     return _sensor.Temperature;
116 }
117
118 private double GetHumDht11()
119 {
120     return _sensor.Humidity;
121 }
122
123 private double GetTempDht22()
124 {
125     return _sensor.Temperature;
126 }
127
128 private double GetHumDht22()
129 {
130     return _sensor.Humidity;
131 }
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
25
```


Jokes : le développeur est un animal bizarre



I Hate Programming
I Hate Programming
I Hate Programming
Oh Sh*t, It Works!!!
I Love Programming

MY CODE
WORKS!
I have no idea why :/



Une publication Nefer-IT, 57 rue de Gisors, 95300 Pontoise - redaction@programmez.com
Tél. : 09 86 73 61 08 - Directeur de la publication : François Tonic
Rédacteurs en chef : François Tonic
Secrétaire de rédaction : Olivier Pavie
Ont collaboré à ce numéro : la rédaction de ZDnet

Nos experts techniques : Finance Active, C. Pellerin, M-L Chamla, A. Gilles, M. Hage Chahine, E. Riou du Cosquer, B. Butel, A. Giretti, P. Boulanger, T. Falque, M. Deremer-Acctone, A. Hanss, J-M Torres, C. Pichaud, J. Lacognata, K. Ansard, P-Y Aillet, L. Ellerbach
Maquette : Pierre Sandré.

Publicité : François Tonic / Nefer-IT - Tél. : 09 86 73 61 08 - ftonic@programmez.com.

Imprimeur : SIB Imprimerie

Marketing et promotion des ventes : Agence BOCONSEIL - Analyse Media Etude - Directeur : Otto BORSCHA oborscha@boconseilame.fr
Responsable titre : Terry MATTARD Téléphone : 09 67 32 09 34

Contacts : Rédacteur en chef : ftonic@programmez.com - Rédaction : redaction@programmez.com - Webmaster : webmaster@programmez.com

Evénements / agenda : redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1220K78366 - ISSN : 1627-0908 - © NEFER-IT / Programmez, avril 2020
Toute reproduction intégrale ou partielle est interdite sans accord des auteurs et du directeur de la publication.

Abonnement :

Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex. - Tél. : 01 55 56 70 55 - abonnements.programmez@groupe-gli.com
Fax : 01 55 56 70 91 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30.

Tarifs

Abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 49 € - Etudiant : 39 € CEE et Suisse : 55,82 € - Algérie, Maroc, Tunisie : 59,89 € Canada : 68,36 € - Tom : 83,65 € - Dom : 66,82 € - Autres pays : nous consulter.

PDF

35 € (monde entier) souscription sur www.programmez.com

HISTOIRE DE LA MICRO-INFORMATIQUE

NOUVELLE CAMPAGNE ULULE

Deux nouveaux volumes :

Volume 2 : **les systèmes d'exploitation**

Volume 3 : **les micro-ordinateurs, +50 nouvelles machines !**

Soutenez-nous : <https://fr.ulule.com/une-histoire-de-la-micro-informatique/>

CAMPAGNE ULULE

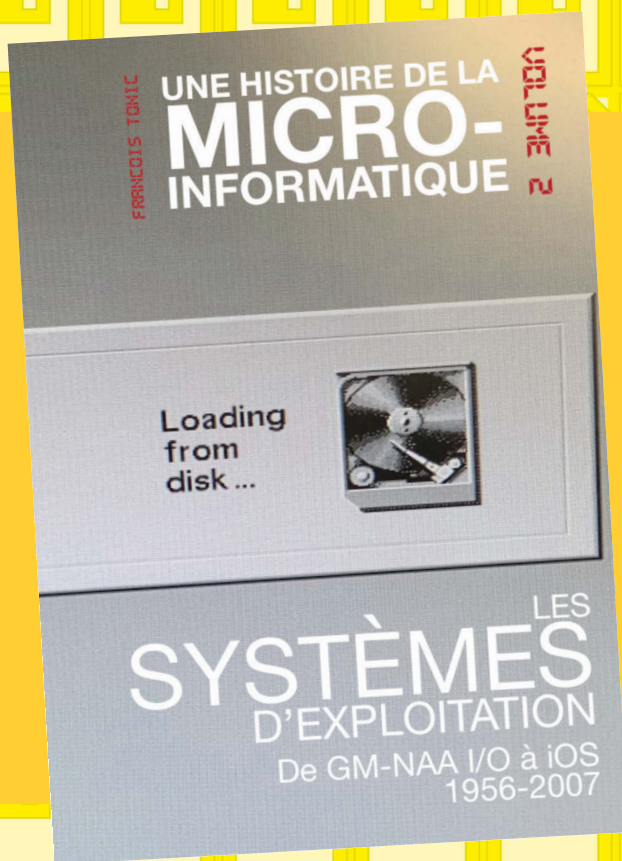
jusqu'au 15 juillet

2 volumes

+200 pages

Format magazine A4 - Type Mook

Sortie prévisionnelle : novembre 2020



Couverture provisoire



A DÉCOUVRIR D'URGENCE

Une histoire de la micro-informatique

Les **ordinateurs** de
1973 à 2007

LE
**CADEAU
GEEK
IDÉAL**

9,99 €
(+ 3 € de frais
postaux)

Découvrez l'âge d'or
des micro-ordinateurs
de 1973 à 2007

[Programmez!]
Le magazine des développeurs

116 pages - Format magazine A4

Disponible sur www.programmez.com