XPERT TECHNOLOGIES

Pourquoi Go ?

Il existe beaucoup de langages pour écrire des applications web. Parmi les plus connus, voici ceux que nous avons évalués mais pas retenus :

- Java : usine à gaz + déploiement chez les clients + stratégie peu claire d'Oracle
- C / C++ : trop compliqués
- PHP / Node.js : trop lents + déploiement chez les clients

Voici les raisons qui nous ont fait opter pour Go :

Compilation

Go est un langage compilé. Cette compilation s'effectue extrêmement rapidement (un grand nombre de type de plateformes cibles peuvent être indiquées). Ainsi, beaucoup de soucis peuvent être évités lors du déploiement (pas de machines virtuelles ou autres dépendances à préinstaller), ce qui est particulièrement appréciable lorsque le déploiement s'effectue chez le client.

Taille

Une fois compilées, les applications ont une taille de quelques mégas, souvent entre 5 et 15 MB (serveur http inclus !).

Performances

Go est un langage jeune (2009) mais a déjà des performances similaires à celles de Java. Son avenir paraît très prometteur.

Stabilité

Au cours de nos différents tests (de charge notamment), l'application ne s'est jamais arrêtée.

Footprint

En plus d'être de petite taille, les applications écrites en Go sont très économes en ressources CPU et en mémoire RAM.

Communauté

Le langage dispose d'une grande communauté. Il est ainsi possible de trouver très facilement des ressources (tutoriaux, livres, vidéos, aide sur Stackoverflow ou autres plateformes) pour la formation ou en cas de problèmes.

Librairies

Go est très populaire et dispose d'un nombre considérable de librairies et driver.

Simplicité

Le langage est très facile à prendre en main. Il ne dispose d'ailleurs que de 25 mots-clés. Il suffit ainsi de quelques heures à un développeur pour être capable de l'utiliser et de quelques jours (~2 semaines) pour le maîtriser.

Multithread

La programmation parallèle est très simple en Go. De plus, le coût d'utilisation des *goroutines* (2kB) est très limité par rapport à l'utilisation de threads (2MB) dans la plupart des autres langages.

Concurrence

Les solutions pour gérer la concurrence est également remarquable avec Go. En plus des *channels* (prônés par l'équipe de Go), il est possible d'utiliser des *mutex* ainsi que des opérations atomiques pour les nombres entiers.

Gestion des erreurs

La gestion des erreurs est plus verbeuse que dans d'autres langages car Go pousse à traiter tous les cas d'erreur (notamment grâce au fait que les procédures peuvent retourner plusieurs résultats). C'est une critique que l'on peut lire quelques fois sur les sites.

En ce qui nous concerne, c'est non seulement une bonne pratique mais ce que tout développeur professionnel devrait faire.

Lisibilité du code

Le code écrit en Go est en règle générale très simple à lire. C'est un point capital pour la maintenance des applications.

Qualité du code

Quelques bonnes pratiques proposées sur le site de Go (qui peuvent pour la plupart être contrôlées à l'aide de l'outil *golint*) ainsi que l'outil *gofmt* pour le formatage permettent d'assurer une excellente qualité du code.

Soutien

Go est soutenu par Google qui est une entreprise sachant ce qu'innover veut dire et propose de fréquentes nouvelles versions. C'est d'ailleurs cette stratégie qui a permis à Chrome de devenir le navigateur le plus utilisé en l'espace de 3 ans.

Une nouvelle version de Go paraît tous les 6 mois et, régulièrement, des correctifs sont proposées (principalement pour des problèmes liés à la sécurité).

Compatibilité des nouvelles versions

Depuis le release 1.0, les nouvelles versions n'ont pas amené d'incompatibilités. Il suffit d'installer la nouvelle version et de compiler son programme pour qu'il marche.

Test & benchmark

L'écrire des tests unitaires en Go est très simple. Tout est intégré au langage.

Il est également possible d'écrire des exemples d'utilisation des fonctions (utilisés comme test unitaire et pris en compte dans la documentation) ainsi que des benchmarks.

Documentation

Si le développeur respecte les bonnes pratiques pour commenter son code, la documentation peut être générée à l'aide d'une simple commande.

POO

Go permet d'utiliser les concepts les plus importants du développement orienté objet, comme le polymorphisme (à l'aide d'interface) au travers de struct (comme en C) plutôt que de classes. Il évite également le concept d'héritage au profit de la composition.
Expérience faite, il semble que ces choix semblent judicieux.

Conclusion

A notre sens, Go est le langage idéal pour le développement backend. Pour notre application principale qui est la réalisation de serveurs Rest, il offre performance et sécurité sans dépendance.

Son côté pragmatique et son code lisible, facile à maintenir, nous ont totalement convaincus.

UPDATE

Go a été nommé langage des années 2009 et 2016 par Tiobe.