

### CTO en un mot

Le CTO identifie les problèmes clés (Loi de Pareto) et essaye de les résoudre. Des problèmes techniques, mais pas seulement.

### Les casquettes du CTO

orga, archi, dev, infra, sécurité, manager, tech lead, recruteur, support, avant-vente

### Les missions du CTO

Gère le quotidien opérationnel

Met en place un environnement qui favorise l'engagement de l'équipe

Organise les interfaces de l'équipe technique

Garantit la bonne exécution technique de la vision produit

Manage les techs

Est rattaché au CEO

Participe au CODIR

A une influence stratégique sur la technologie

Gère le budget

Définit la vision technologique

Supervise la RD, l'innovation

Fait le sale boulot

Est responsable de la sécurité (et de la conformité)

Arbitre, priorise pour aligner les choix techniques, produit et business

Est responsable de la documentation

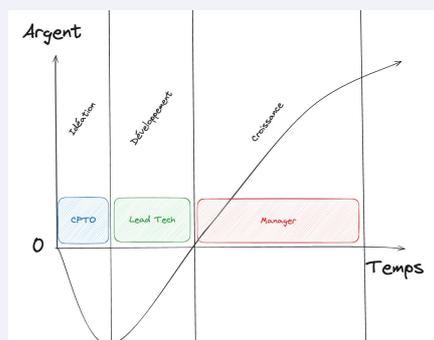
Gère l'amélioration continue

Utilise des indicateurs (DORA, uptime, etc.) en faisant attention à la loi de

Communique vers ses équipes et informe l'extérieur

Pilote les post mortem

### Le but final du CTO



Le but final du CTO est de prendre en charge des sujets, automatiser, scaler, déléguer et passer sur d'autres sujets.  
Crédit graph Hugo Dupré

### Recruter en 4 étapes

1e étape : Entretien de présentation et fit humain avec le CTO

2e étape : Entretien tech par un membre de l'équipe tech

3e étape : 1 journée immersion payée

4e étape : Faire une proposition

Bonus : ne pas ghoster. En cas d'arrêt du process, faire un retour honnête

L'exercice de l'entretien classique ne teste pas si le candidat convient au poste visé, il teste si le candidat est bon en entretien. Évitez le double écueil de favoriser les candidats optimisés pour les entretiens et de défavoriser les bon tech plus faibles en entretien.

### Fidéliser un tech

Salaire au niveau du marché ou supérieur

Augmentation annuelle

TT ou espace coworking ou présentiel libre

Individualisation des leviers de motivation et avantages

Haut niveau d'exigence

### Créer et maintenir l'engagement de l'équipe

Célébrer les victoires

Faire des one to one réguliers

Utiliser la méthode américaine (inspirée de l'éducation positive)

Éviter absolument les injustices

Manager en remote demande des compétences spécifiques qu'il faut développer

### Agile... mais pas trop !

Les individus et leurs interactions plus que les processus et les outils... mais les processus et outils sont importants

Des logiciels opérationnels plus qu'une documentation exhaustive... mais la doc reste importante !

La collaboration avec les clients plus que la négociation contractuelle... mais la négociation contractuelle reste importante

L'adaptation au changement plus que le suivi d'un plan... mais les specs détaillées sont primordiales

Bonus : le leadership, l'exemplarité, l'intelligence collective à la place du management directif/hierarchique

### Comment résoudre un problème

N'importe quel problème complexe peut être découpé en plus petit problèmes plus simple à résoudre

Isoler le problème : simplifier au maximum le problème pour n'avoir qu'un paramètre à traiter

Essai/erreur : émettre une hypothèse, tester la solution, vérifier le résultat, itérer



### Security by design

Minimiser la surface d'attaque

Appliquer le principe de moindre privilège

Utiliser une approche défensive au niveau du code

Bonus : Mitiger les attaques par force brute (page de login, DDOS, etc.)

### Rembourser la dette technique

Funday : un jour par semaine de refacto

La Guerrilla contre les bugs : 30% des évolutions sur du refacto

Le Drynuary : un mois entier de remboursement de la dette technique

Atelier quickwin d'une heure en pair avec le produit

### Bonnes pratiques design logiciel

KISS

SOLID

9 règles des objets Calisthenics

fail fast, modes strictes, typage fort

Mettre à jour la stack et les dépendances et utiliser les innovations

Séparer front et back et passer en API

Supprimer le code mort, supprimer les features inutilisées

Créer et utiliser des value objects significatives plutôt que les primitives

Tendre vers le DDD

Tendre vers l'architecture hexagonale

Passer en TDD

Extraire les parties plus basses ou non métiers, utilisables dans d'autres projets dans un projet séparé open source qui devient une dépendance.

Paralléliser les TU

Traiter les requêtes n+1

Rendre le système observable et monitoré

### Les événements de l'équipe

1 fois par mois : retrospective entre techs (amélioration de nos process, évocation des bloqueurs, gros sujets à aborder etc.)

1 fois par mois : Le CTO fais un one to one avec chacun des techs

lundi : priorisation/planification de ce qu'on va faire dans la semaine

1 fois par jour : daily rapide pendant laquelle on fait une revue des tickets en cours et on se pose un peu plus sur les tickets qui n'avancent pas (qui sont resté dans le même status plus de 24h ou 48h). Mais pas de tour de table systématique. Et chacun peut aussi annoncer un atelier qu'il organise sur une problématique qu'il synthétise.

Des ateliers proposées à l'initiative des techs sur lesquels les autres techs peuvent s'inscrire.

Et surtout, beaucoup beaucoup d'asynchrone via Slack

