

# Comment une entreprise SaaS peut-elle veiller à être parée pour l’avenir ?

18 octobre 2022, 14:49

Nick Boucart

**En ma qualité d’expert Sirris, je reçois régulièrement des questions d'entrepreneurs SaaS à propos de problèmes techniques. Il s’agit là d’une réalité quotidienne pour de nombreuses start-ups et scale-ups. Les réponses et les solutions que je propose peuvent donc être utiles à d'autres entreprises également. Voici quelques exemples récents de telles questions et de la façon d’y faire face.**

## « Je me demande si nous faisons bien ce qu'il faut. »

C’est l'un des fondateurs d'une scale-up SaaS prometteuse qui parle : « J'ai remarqué que nous ne progressions pas aussi rapidement ces derniers temps. Nos clients trouvent régulièrement de nouveaux bugs et, honnêtement, je me demande parfois si nous utilisons la bonne technologie pour notre plate-forme. »

On me pose assez souvent ce genre de questions. Créer un SaaS n'est pas chose simple, et la relation entre le business (ventes, marketing, clients) d'une part et l'équipe technique qui doit tout construire d'autre part est parfois très tendue. Les premiers préfèrent des tonnes de fonctionnalités, ici et maintenant, alors que les développeurs sont souvent immergés au plus profond de leur code.

*Deux univers en collision*

Quand on me pose une telle question, je propose généralement de réfléchir avec l'équipe pendant une demi-journée. Je pose alors des questions ouvertes sur la façon dont l'équipe travaille, sur les technologies qu'elle utilise et, surtout, sur la manière dont elle les utilise. Comment les exigences sont-elles définies ? Comment est établi le planning ? Des tests unitaires ou d'autres tests (automatisés) sont-ils développés ? etc.

Presque toujours, après un tel entretien, je parviens à formuler 4 à 5 recommandations concrètes pour rapprocher le business et l'équipe technique, pour donner une impulsion qui permettra d'améliorer la confiance entre le business et le développement. Au bout du processus, du moins en ce qui me concerne, plus le business et l'équipe technique se comprennent, plus ils connaissent les sensibilités et le travail de l'autre, plus les chances de réussite sont grandes.

## « Comment faire pour industrialiser notre logiciel ? »

« Nos clients exécutent beaucoup plus de simulations qu'auparavant et notre serveur a du mal à suivre, explique le responsable R&D d'une société d'ingénierie qui a connu une croissance rapide ces dernières années et qui a traduit une partie de son savoir-faire en un produit SaaS.

Nos ingénieurs font des études dans notre domaine et ils développent et affinent constamment nos algorithmes et nos méthodes de calcul. Ces scripts Python servent de base à notre partenaire technique, qui les intègre dans une application web/plate-forme SaaS que nous commercialisons avec succès depuis un certain temps. Ce que nous constatons aujourd'hui, c'est que notre architecture et notre infrastructure actuelles ont de plus en plus de mal à exécuter nos simulations en temps voulu. Nous cherchons des moyens de les rendre plus robustes. Le cloud peut-il nous y aider, et si oui, comment faire ? »

Ce n'est pas la première fois qu'on me pose cette question. Les équipes techniques s'intéressent de plus en plus souvent au cloud comme solution potentielle afin de mieux faire évoluer leur plate-forme SaaS. Mais honnêtement, il n'est pas aisé de faire le pas vers le cloud. Il suffit de jeter un coup d'œil aux pages produits d'AWS ou d'Azure pour être inondé de jargon, de listes de services, d'études de cas auxquels un simple mortel peut rarement s'identifier et de pages de prix dont la compréhension exige des études supérieures en mathématiques. Pour s'y retrouver et avoir une idée de ce à quoi pourrait ressembler une combinaison de différentes options de calcul et de stockage pour votre situation spécifique, il faut souvent passer par une sérieuse courbe d'apprentissage.

Habituellement, j'utilise des exemples d'architectures pour esquisser différentes possibilités de traitement d'un tel problème, puis je lance une réflexion sur les implications de ces choix. Je passe ensuite en revue les avantages et les inconvénients de certains choix techniques avec l'équipe, en veillant à ce qu'elle prenne en compte les aspects opérationnels tels que le suivi, le dimensionnement (automatique) et la sécurité. Au cours de ce processus, il s'avère souvent que l'équipe technique n'est pas très au fait des tendances telles que le « sans serveur » ou les conteneurs, et qu'elle ne peut donc pas évaluer ce que cela pourrait signifier pour elle.

Dans le cas ci-dessus, nous avons utilisé une démonstration de faisabilité pour voir comment AWS Lambda, AWS S3 et AWS Step Functions pouvaient rendre le moteur de calcul beaucoup plus modulaire et transparent. En travaillant sur une démonstration de faisabilité pratique, l'équipe a également pu se familiariser avec des aspects tels que IAM, « infrastructure-as-code », AWS Cloudwatch, etc. Autant d'éléments qu'elle rencontrera de toute façon tôt ou tard, mais qui ont

tendance à rendre malaisés les premiers pas dans le cloud.

### S'organiser pour le changement

Lorsque je parle à l'équipe de développement d'une start-up ou d'une scale-up, j'essaie toujours de savoir comment elle s'organise pour le changement. S'il y a bien une constante dans le domaine des logiciels, c'est le changement. Chaque contact avec des clients, des prospects ou des utilisateurs apporte de nouvelles idées pour améliorer le produit, de nouveaux membres rejoignent l'équipe ou quelqu'un la quitte, de nouvelles briques de cloud computing/open-source font apparaître de nouvelles opportunités. La question n'est donc pas tant « qu'est-ce qui va encore changer demain ? », mais plutôt « comment pouvons-nous organiser le développement de nos logiciels de manière à pouvoir faire face au mieux aux changements de demain ? »

Par exemple, je demande toujours d'expliquer en termes généraux l'architecture du logiciel. Ma règle d'or pour être « à l'épreuve du changement » est la suivante : « the simplest solution that could possibly work ». Même si elle ne fait rien de mal, une équipe de trois personnes qui construit des microservices ne se facilite pas la tâche, surtout qu'en pratique, la plupart des personnalisations exigent aussi l’adaptation de chaque microservice.

Les tests automatisés sont un autre outil qui permet de mieux faire face au changement. Il peut avant tout s’agir de tests unitaires, ou éventuellement de quelques tests fonctionnels et/ou de bout en bout. J'accorde une grande importance aux tests automatisés pour plusieurs raisons :

- Ils constituent un filet de sécurité pour les développeurs : une bonne couverture des tests unitaires leur permet de savoir rapidement si ce qui fonctionnait hier fonctionne encore aujourd'hui.
- Les tests automatisés sont également une forme de documentation : ils montrent comment le code fonctionne et peut être utilisé, maintenant et à l'avenir.

Pour moi, l'automatisation des tests permet à la fois de maintenir le code existant et de rendre l'équipe à l'épreuve des changements.

Vous vous reconnaissez dans une telle situation ? Je suis toujours disposé à échanger des idées avec vous !

Consultez également notre Masterclass sur la cybersécurité - SaaS, plateformes numériques, logiciels!

## Auteurs



Nick Boucart