

Six pièges avant de brancher un agent IA à son CRM

Un agent écrit vite, donc il se trompe vite. Voici les pièges que j'ai rencontrés, chacun avec ce qui se passe et le réflexe qui l'évite.

1. Le code 200 n'est pas une preuve

CE QUI SE PASSE

HubSpot répond « 200, c'est bon » quand on écrit le statut marketing d'un contact par l'API. La propriété est en lecture seule : rien ne change.

QUOI FAIRE

On relit la valeur après chaque écriture et on la compare à ce qu'on attendait. Le code de retour ne compte pas comme preuve.

LE RÉFLEXE

Quand une propriété est en lecture seule, on passe par une liste statique et l'interface, puis on relit le statut.

2. La lecture en lot qui ignore les associations

CE QUI SE PASSE

La lecture en lot des contacts (`batch/read`) ignore le paramètre `associations` sans erreur ni avertissement. Un garde-fou du type « sauter les contacts qui ont déjà une entreprise » ne trouve alors aucun cas et laisse tout passer.

QUOI FAIRE

On lit la propriété `associatedcompanyid` , qui passe en lot et porte l'entreprise principale.

LE RÉFLEXE

Tout garde-fou se teste sur un cas qu'il doit bloquer. Un garde-fou qui ne bloque jamais est peut-être inerte.

3. La donnée qui arrive en retard

CE QUI SE PASSE

`associatedcompanyid` se met à jour avec un retard, après une écriture comme après un retrait. Une relecture immédiate annonce un échec qui n'en est pas un.

QUOI FAIRE

On attend (de quelques secondes à quelques minutes) et on relit avant de rejouer l'opération. Rejouer trop tôt crée des doublons.

LE RÉFLEXE

Une vérification qui échoue une première fois se refait quelques minutes plus tard avant de conclure.

4. Le type d'email qu'on ne peut plus changer

CE QUI SE PASSE

Un email créé en mode « lot » ne devient jamais un email de workflow. Le type est figé à la création, le chemin du gabarit aussi.

QUOI FAIRE

On clone un email déjà « automatisé » (`POST /marketing/v3/emails/clone`), puis on remplace son contenu.

LE RÉFLEXE

On crée le bon type dès le départ, et on le note dans la fiche de la campagne.

5. L'accolade qui efface la page

CE QUI SE PASSE

Dans le code d'une page, `{#` ouvre un commentaire du langage de gabarit HubSpot. Une règle de style écrite `{#mon-id` avale la fin de l'en-tête : la page s'affiche blanche, alors que le HTML brut a l'air complet.

QUOI FAIRE

On écrit `{ #mon-id`, avec une espace après l'accolade.

LE RÉFLEXE

Avant toute mise à jour, on cherche `{#` dans le texte envoyé. Après, on regarde la page rendue, pas seulement le HTML.

6. La production et le bac à sable

CE QUI SE PASSE

Un agent qui peut écrire n'importe où écrira un jour au mauvais endroit.

QUOI FAIRE

Deux comptes séparés, et un garde-fou mécanique (un hook) qui refuse toute écriture en production sans autorisation explicite.

LE RÉFLEXE

Le bac à sable sert aux essais. La production ne reçoit que ce qui a déjà été vu ailleurs.

Le protocole avant de laisser un agent écrire

1. **Lecture seule d'abord.** L'agent lit, résume et propose. Il n'écrit pas encore.
2. **Un compte d'essai.** Les premiers tests se font ailleurs qu'en production.
3. **Un petit lot.** On écrit sur cinq enregistrements, pas sur cinq mille.
4. **On relit l'effet réel.** La valeur est-elle bien changée, ailleurs que dans la réponse de l'API ?
5. **Un journal et un retour arrière.** Chaque écriture est notée, et on sait comment l'annuler.

Ces six pièges viennent de vrais branchements d'agents sur un CRM HubSpot, notés au moment où ils sont apparus. Les comportements d'une API peuvent changer : refaites le test sur votre compte avant de vous y fier.

Clément Cathala, GTM Engineer. Fiche gratuite, à copier, adapter et partager. clementcathala.com