RAPPORT SUR L'ARCHITECTURE IA ET LA STRATÉGIE D'ENTREPRISE

L'effet mouche : architectures d'agents événementielles et l'illusion du coût

Réduire les coûts des LLM en entreprise, à partir du principe d'inertie dans une voiture en mouvement

Mehmet Keskinkılıç · Novembre 2026 **Catégorie :** Conception de systèmes et optimisation financière

Approche : Machines à états déterministes + inférence ciblée

Résumé : *De nombreuses entreprises repoussent leur transformation par l'IA et l'intégration d'agents autonomes à cause de « coûts d'API imprévisibles ». Une cause importante de cette réticence n'est pas un manque de technologie, mais une architecture qui déclenche de grands modèles à chaque étape du flux de travail. Inspiré de la physique d'une mouche dans une voiture fermée, ce document propose une feuille de route pour un « modèle d'agent à accélération » : les opérations courantes restent à zéro jeton et le LLM ne tourne qu'en cas d'anomalie. Les chiffres de ce rapport sont des hypothèses de scénario d'exemple, pas des mesures.*

1. La métaphore physique : pourquoi la mouche n'est-elle pas projetée dans l'habitacle fermé ?

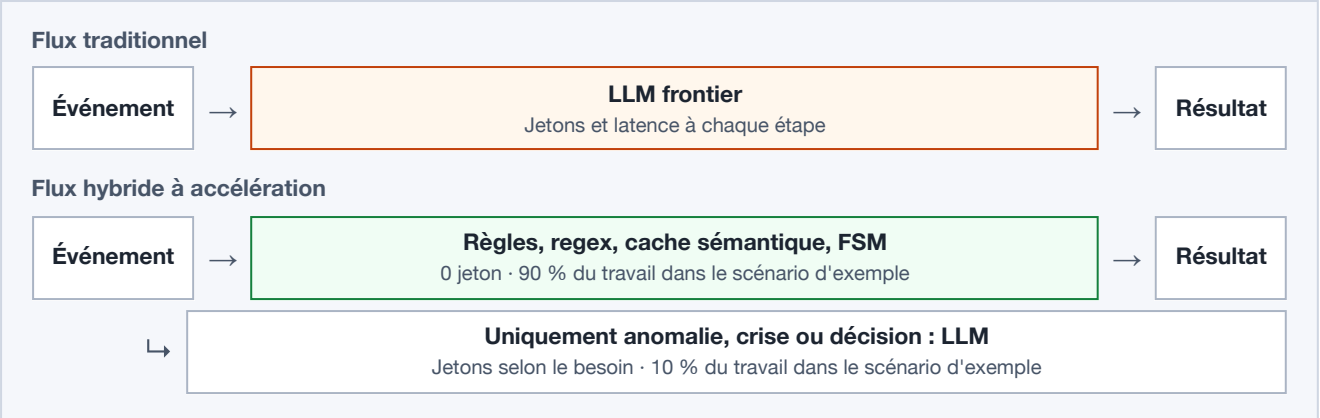
Une mouche dans une voiture fermée roulant à 100 km/h ne fait aucun effort supplémentaire pour rester en vol stationnaire ou voler du pare-brise à la banquette arrière. Il y a deux raisons physiques : *la masse d'air transportée à la même vitesse* et le principe d'*inertie*. La mouche se déplace par rapport à l'air immobile de l'habitacle, non par rapport à la route. Elle ne dépense beaucoup d'énergie pour garder son équilibre que si la voiture freine brusquement, accélère ou si une fenêtre ouverte laisse entrer des turbulences.

Les conceptions d'agents traditionnelles se comportent comme si la mouche devait voler hors de la voiture, contre un vent de 100 km/h : chaque saisie courante, chaque contrôle d'état et chaque simple validation déclenche un grand modèle de langage (LLM) du début à la fin, et la consommation de jetons devient incontrôlée.

2. L'erreur en entreprise : le réflexe de brancher un LLM à chaque étape

Derrière l'idée que « la transformation par l'IA dépassera notre budget », on trouve souvent une conception d'agent monolithique qui transmet chaque événement opérationnel à un modèle frontier coûteux. Pourtant, une part importante des processus d'entreprise (demandes clients, vérifications de factures, mises à jour ERP, routage de tickets) peut s'exécuter avec des règles prévisibles et déterministes. Cette part varie selon le processus et doit être mesurée ; le scénario d'exemple de ce rapport suppose que 90 % du travail se termine sans atteindre un modèle.

© 2026 Mehmet Keskinkılıç · mehmetkeskinkilic.com



3. Les trois piliers de l'architecture

A. Couche de filtrage déterministe (Gating Layer)

Chaque entrée passe d'abord par des contrôles à coût nul ou minime. Si elle correspond à un modèle standard, à une route d'API prédéfinie ou à une validation par règles, l'opération se termine directement dans le code. Aucun appel d'API LLM n'est effectué et aucun jeton n'est consommé.

B. Flux de travail fondé sur une machine à états (Finite State Machine, FSM)

Les processus de l'agent sont modélisés avec des cadres orientés états comme LangGraph. Lors du passage d'un état à l'autre, les règles de décision s'exécutent dans le code. Seules une exception inattendue, une anomalie de données non structurées ou une bifurcation stratégique à plusieurs options font lever au système une alerte d'« accélération » et le dirigent vers le nœud LLM.

C. Cache sémantique et inférence par différences (Diff)

Les requêtes répétées sont repérées par de légers modèles d'embedding locaux et servies directement depuis le cache. Même lorsqu'un appel au LLM est nécessaire, tout le corpus documentaire de l'entreprise n'est pas renvoyé à chaque fois ; le prompt caching et l'envoi des seuls paramètres d'état modifiés (le delta) réduisent les dépenses.

4. Comparaison des coûts et des performances (scénario d'exemple)

Le tableau ci-dessous n'est pas la mesure d'un déploiement réel. C'est un scénario d'exemple pour un centre de support ou de traitement de taille moyenne gérant 10 000 événements opérationnels par jour.

Indicateur / Étape	Agent LLM monolithique	Modèle hybride à accélération	Économies et effet
Flux courant (75 %)	~1 500 jetons par appel	0 jeton (exécution au niveau des règles et de l'API)	100 % de jetons LLM économisés
Entrée similaire ou répétée (15 %)	Le modèle raisonne depuis le début	0 jeton (correspondance dans le cache sémantique)	100 % de jetons LLM économisés
Anomalie et décision (10 %)	Fonctionne avec un contexte coûteux	Petit modèle ou modèle optimisé + contexte diff minimal	70 % d'optimisation des jetons
Temps de réponse d'exemple	1 800 ms – 3 500 ms	< 40 ms (courant) / 900 ms (anomalie)	Plus de 97 % en moins en courant, environ 50–75 % en moins sur les anomalies
Coût mensuel estimé	4 500 $ – 7 000 $ / mois	350 $ – 600 $ / mois	**Environ 87 % – 95 % d'économies estimées**

Hypothèse de calcul : environ 300 000 événements par mois × ~1 500 jetons = ~450 millions de jetons, à environ 10–16 $ en moyenne par million de jetons. Sur cette base, la part de 10 % d'anomalies du modèle hybride coûte environ 135–215 $ en jetons ; la fourchette du tableau se situe au-dessus, par prudence. L'écriture des règles, l'infrastructure, la maintenance et l'exploitation du cache ne sont incluses dans aucune des deux colonnes.

5. Synthèse pour le conseil d'administration et plan d'action

> **Conclusion stratégique :** La transformation par l'IA ne consiste pas à « confier tous les processus à un LLM ». Une transformation réussie sépare, aux bons seuils, la fiabilité déterministe du génie logiciel et la capacité de résolution de problèmes des LLM. Lorsque, comme dans le scénario d'exemple, 90 % du travail s'exécute sans consommer de jetons et que l'IA n'intervient que dans les 10 % de moments incertains restants, la barrière du coût peut devenir un avantage.

Plan d'action

1. **Mesurer :** Enregistrez les appels LLM actuels par type d'événement, avec volume, jetons, latence et taux d'erreur. Remplacez les proportions de ce rapport par vos propres données.
2. **Séparer :** Déplacez le travail courant à fort volume et à résultat vérifiable vers des règles et des machines à états.
3. **Ouvrir le cache sous contrôle :** Utilisez le cache sémantique avec un seuil de similarité, contrôlez des échantillons et empêchez les données personnelles ou sensibles de passer d'une réponse à l'autre.
4. **Fixer une règle d'escalade :** Envoyez d'abord les anomalies à un petit modèle. Si la confiance est faible, passez à un grand modèle ou à une personne.
5. **Surveiller en continu :** Suivez la part du travail réalisée sans modèle, les réponses erronées des règles et du cache, et le coût mensuel total, ingénierie comprise.

Mehmet Keskinkılıç
Spécialiste des opérations aériennes & développeur iOS indépendant
© 2026 Mehmet Keskinkılıç. Partage autorisé avec mention de la source. Version à jour : mehmetkeskinkilic.com/fr/writing/