CONCEPTION PRODUIT · MÉMOIRE DES AGENTS

Du souvenir au pouvoir d'agir

Comment un agent à modèle figé peut accumuler de l'expérience sans laisser son passé décider à sa place

LE POINT DE DÉPART

Depuis plusieurs mois, je construis Cortex.deck autour d'un problème simple à formuler : comment faire pour qu'un agent ne redécouvre pas tout à chaque nouvelle mission ?

Au départ, je cherchais surtout à conserver du contexte. J'ai progressivement dû distinguer une conversation, une trace, une expérience, un apprentissage et un souvenir. Puis une difficulté plus importante est apparue : dès qu'une information passée peut modifier une décision future, la mémoire reçoit un pouvoir.

Ce texte raconte ce chemin depuis le terrain de la conception produit. Il traite la mémoire d'un agent comme un problème de système : ce qui est capté, relié, reconstruit, autorisé et finalement utilisé. Les rapprochements avec la mémoire humaine servent à rendre le problème visible ; ils ne décrivent pas un modèle du cerveau.

Les travaux cités à la fin situent cette construction parmi les recherches sur la mémoire des agents, la révision de croyances et la provenance. Le texte distingue ce que Cortex matérialise déjà de la chaîne qu'il reste à fermer.

Introduction — Une mémoire ne devient pas une expérience parce qu'elle persiste

Hier, quelqu'un m'a affirmé savoir ce que j'avais dit trois semaines plus tôt.

Je me souvenais de la conversation. J'avais partagé une intuition : un élément finirait forcément par avoir un impact à un moment donné. Lui se souvenait d'une formulation plus forte : « Tu avais dit que cela allait tout casser. »

Cette version me paraissait exagérée. Comme il consultait ses notes, je lui ai demandé si la phrase y apparaissait. Non. Il en avait le souvenir.

J'aurais aimé qu'une note existe, mais pas seulement pour déterminer lequel de nous deux avait raison. Elle aurait permis d'observer la fabrication de nos souvenirs. Si elle indiquait « cet élément aura forcément un impact », nous aurions vu comment sa mémoire avait transformé

une intuition en rupture annoncée. Si elle contenait réellement « cela va tout casser », elle aurait révélé que mon propre souvenir avait atténué mes mots. Une note ambiguë aurait montré comment chacun de nous avait rempli les espaces avec ce qui s'était produit ensuite.

Le plus intéressant aurait été de suivre les conséquences. Qu'avait-il remarqué, ignoré ou décidé pendant trois semaines parce qu'il pensait que j'avais annoncé une rupture ? Une note nuancée aurait peut-être limité cette interprétation. Une note déjà excessive aurait pu, au contraire, l'ancrer et transformer chaque nouvel événement en confirmation.

Cette discussion résume une grande partie du problème que je rencontre avec les agents.

Une trace possède deux vies. Il y a le moment où elle est captée, puis les moments où elle revient dans une nouvelle situation. Entre les deux, elle peut être résumée, rapprochée d'autres événements, interprétée, contredite ou privée de son contexte initial. Lorsqu'elle revient, elle ne conserve plus seulement le passé. Elle devient une cause possible du futur.

> Une mémoire fiable n'est donc pas une mémoire qui ne se trompe jamais.

Une telle promesse serait intenable, pour un humain comme pour un logiciel. Elle doit au moins savoir distinguer ce qui a été observé, ce qui en a été déduit, ce qui reste incertain et la manière dont ces éléments ont été utilisés ensuite.

C'est mon point de départ. Conserver ne suffit pas. Retrouver ne suffit pas non plus. Il faut comprendre comment la matière du passé acquiert, conserve ou perd le droit d'influencer le présent.

01

Le modèle n'est pas l'agent

Lorsque nous parlons d'intelligence artificielle, nous confondons souvent le modèle et l'agent. Pourtant, ce ne sont pas les mêmes objets.

Un modèle comme GPT, Claude, Gemini ou GLM est un moteur de génération et de raisonnement. Il reçoit un contexte, produit une réponse, puis s'arrête. Dans l'usage courant de ces services, ses paramètres ne changent pas parce qu'une conversation s'est bien ou mal passée. Il peut corriger sa réponse dans le même échange, mais cette correction ne devient pas automatiquement une expérience disponible deux mois plus tard.

L'agent est le système construit autour de ce moteur. Il possède des outils, des permissions, des fichiers, des règles, une mémoire et parfois la capacité de provoquer des effets en dehors de la conversation. Le même modèle peut ainsi participer à des agents très différents.

Donnez-lui un autre contexte, d'autres outils ou d'autres droits et son comportement observable change, même si son moteur reste identique.

Séparer le modèle de l'agent m'a permis de sortir d'une fausse impasse. Si le modèle est figé, tout le système ne l'est pas forcément. L'expérience peut appartenir à l'agent plutôt qu'au modèle. Des travaux comme Reflexion et ExpeL ont déjà montré qu'un système pouvait réutiliser des retours d'expérience sans modifier les paramètres du modèle principal [1][2]. Mon problème n'était donc pas de réentraîner une intelligence artificielle après chaque mission. Il était de construire autour d'elle une continuité suffisamment précise pour modifier les missions suivantes.

Il faut néanmoins être prudent avec le mot « apprendre ». Une base de données qui grossit n'apprend pas. Un historique qui s'allonge n'apprend pas davantage. J'emploie ici ce mot dans un sens fonctionnel : le système apprend d'une expérience lorsque cette expérience produit une modification durable et identifiable de sa manière d'aborder une famille de situations futures.

Cette définition contient déjà plusieurs exigences. La modification doit survivre à la session. Elle doit pouvoir être reliée à une expérience identifiable. Elle doit avoir un périmètre, car une leçon utile dans un projet peut devenir absurde dans un autre. Elle doit aussi pouvoir être corrigée si la première interprétation se révèle fausse.

Au début, je résumais cette ambition par une image. Je ne cherche pas un stagiaire brillant qui redécouvre tout chaque matin. Je cherche un collaborateur qui accumule de l'expérience.

> Écrire « tu as dix ans d'expérience » dans un prompt ne lui donne pas dix ans d'expérience. Cela lui donne un rôle.

L'expérience se construit avec des situations vécues, des décisions, des résultats, des erreurs et des corrections. Elle ne compte que si elle change durablement la façon d'agir.

Cette continuité ne dépend pas nécessairement d'un seul modèle. J'ai déjà utilisé plusieurs moteurs dans une même session Cortex. Ils ne raisonnaient pas de la même manière, mais l'environnement, les sujets, les règles et la matière disponible restaient ceux d'Arka. Le moteur changeait. Le système conservait son histoire.

Cette continuité n'est pas une identité stable entre modèles. Elle indique où je situe l'expérience : dans les objets, les relations et les règles qui survivent au moteur du moment.

02

Pourquoi une conversation enregistrée ne suffit pas

La première forme de mémoire paraît évidente : conserver les échanges. Beaucoup d'agents disposent donc d'un fichier `MEMORY.md`, d'un historique complet ou d'un résumé régulièrement mis à jour.

Cette solution fonctionne au début. Quelques décisions, préférences et rappels tiennent dans un document lisible. L'agent peut le relire au lancement d'une nouvelle session et donner l'impression de reprendre le fil.

La limite apparaît avec le temps.

Imaginez que vous preniez des notes après chaque réunion, incident et post-mortem pendant une année. Deux mois plus tard, au démarrage d'un nouveau projet, allez-vous relire l'ensemble de vos carnets avant de répondre à la première question ? Probablement pas. Vous cherchez quelques éléments utiles, souvent parce que la situation présente réveille le souvenir d'une ancienne.

Pourtant, injecter un fichier mémoire complet dans le contexte revient à demander exactement l'inverse à l'agent. Il relit tout. Les informations importantes se retrouvent au milieu de décisions dépassées, d'exceptions locales, de formulations approximatives et de détails qui ne concernent pas la mission actuelle.

Le coût en unités de texte, les *tokens*, n'est qu'une partie du problème. Le bruit modifie aussi le raisonnement. Une phrase ancienne peut sembler pertinente parce qu'elle partage quelques mots avec la situation. Une règle écrite pour un projet peut être appliquée ailleurs. Une hypothèse non vérifiée peut revenir sous la forme d'une certitude, simplement parce que le document ne conserve plus son statut initial.

L'autre difficulté vient de l'écriture. Dans mes premières architectures, le modèle principal pouvait modifier lui-même ses fichiers. Cette autonomie paraît séduisante : l'agent découvre quelque chose, puis met sa mémoire à jour. Mais la même opération peut reformuler l'existant, écraser une nuance ou supprimer la matière qui expliquait pourquoi une ancienne conclusion avait paru crédible.

> **Une mémoire que le même modèle peut réécrire librement ressemble à un carnet dans lequel le témoin, l'enquêteur et l'archiviste seraient la même personne.**

Chaque correction peut améliorer le récit. Elle peut aussi rendre l'erreur impossible à reconstruire.

J'ai alors séparé deux responsabilités : l'agent principal réalise la mission ; d'autres composants captent et organisent ce qui s'est passé. Le modèle chargé d'agir n'est plus seul à décider de ce qui mérite d'être retenu. Il peut être une source d'interprétation, mais il n'est pas l'autorité qui transforme cette interprétation en mémoire active.

Ce que j'appelais au départ « l'observateur » s'est progressivement décomposé en plusieurs *workers*, de petits composants spécialisés. Certains captent les échanges et les appels d'outils. D'autres produisent le roulement de chaque tour, c'est-à-dire une représentation courte de ce qui vient de se passer et de ce qui reste ouvert. D'autres encore maintiennent la continuité de la session ou analysent la matière après coup.

Le choix de petits modèles pour une partie de ce travail n'est pas seulement économique. Il sépare les rôles. Le modèle principal peut consacrer sa capacité à la mission, tandis que des modèles moins coûteux exécutent des tâches plus bornées : extraire, classer, vérifier une forme ou préparer une consolidation. LightMem étudie lui aussi cette séparation entre traitement immédiat et consolidation différée à l'aide de petits modèles [3].

Cette chaîne produit une matière plus exploitable qu'un historique. Elle sait que quelque chose a été dit, qu'un outil a été utilisé et qu'un résultat est apparu. Pour obtenir une expérience, il faut encore relier ces morceaux sans inventer ce qui manque.

03

De la note au graphe : conserver les relations, pas seulement les éléments

Après les fichiers Markdown, je suis passé au JSON. Le JSON est un format texte qui range les informations avec des étiquettes. Au lieu d'écrire une phrase au milieu d'un paragraphe, on peut indiquer qu'un élément possède un nom, une date, un type ou une source.

Le gain a été immédiat. Un programme sait où trouver une information et peut vérifier sa forme. Mais le problème de fond revient vite : le contenu reste dispersé dans des fichiers, et les liens entre les éléments sont souvent de simples identifiants. Déplacer, supprimer ou réécrire une partie peut fragiliser tout ce qui y fait référence.

Il me fallait une base de données. J'ai choisi SurrealDB parce qu'elle peut fonctionner directement dans Cortex et réunir plusieurs besoins : conserver des données structurées, effectuer des recherches par mots ou par proximité de sens et représenter un graphe [13].

Le mot « graphe » peut sembler technique. L'idée est pourtant assez simple : les relations deviennent des objets aussi importants que les informations qu'elles relient.

Prenons une enquête policière. Un témoin affirme avoir vu une voiture rouge quitter les lieux à 22 heures. Les enquêteurs identifient son propriétaire et formulent une première hypothèse : cette personne pourrait être impliquée.

Le lendemain, une caméra montre que la plaque a été mal lue. Une autre preuve place le propriétaire ailleurs au même moment.

Dans un fichier, on pourrait remplacer l'ancienne conclusion par « cette personne n'est plus suspecte ». La mémoire contiendrait alors la bonne conclusion, mais elle aurait perdu une partie de l'enquête. Pourquoi cette personne avait-elle été soupçonnée ? Quelle information semblait crédible ? Qu'est-ce qui a invalidé la piste ?

Dans un graphe, le témoignage peut rester relié à l'hypothèse par « soutient ». La vidéo peut être reliée à la même hypothèse par « contredit ». L'alibi peut être relié par « invalide ». Chaque relation possède une origine, une date et un statut. La conclusion actuelle reste accessible sans effacer le chemin qui avait produit l'erreur.

C'est la cible dans Cortex. Lorsqu'une nouvelle preuve apparaît, le système ne doit pas réécrire silencieusement le passé. Il ajoute un élément, crée ou modifie des relations et recalcule ce qui peut encore être conclu.

> Un graphe peut très bien relier des affirmations fausses avec une grande précision.

La base de données ne garantit rien toute seule. Ce sont les règles d'écriture et de transition qui comptent : qui peut créer un élément, sur quelle source, avec quel statut, et dans quelles conditions une relation peut-elle changer le pouvoir d'une croyance ?

Zep, APEX-MEM ou Kumiho explorent eux aussi des architectures temporelles capables de conserver les évolutions, les conflits et les versions successives [4][5][6]. Dans la vision de Cortex, le graphe fournit la matière nécessaire à la suite : une histoire assez structurée pour pouvoir être reconstruite.

À ce stade, Cortex peut disposer d'une carte : échanges, décisions, actions, résultats et relations connues entre eux. Une carte, même détaillée, n'est pas encore une expérience.

04

À quel moment une suite d'échanges devient-elle une expérience ?

Une conversation commence lorsqu'une interface ouvre un chat et se termine lorsqu'on le ferme. Le travail suit rarement cette frontière.

Une même conversation peut contenir plusieurs problèmes indépendants. À l'inverse, une difficulté peut traverser plusieurs sessions, outils et documents. Utiliser le chat comme unité d'expérience revient donc à laisser l'interface décider où commence et où s'arrête le sens.

J'ai trouvé une image plus utile dans les séries.

Chaque tour de conversation ressemble à une scène. Une première scène fait apparaître un problème. Une autre pose l'intrigue. Une décision est prise, une action suit, puis une scène ultérieure montre le résultat. Regardée seule, la scène du milieu peut sembler claire. Elle perd pourtant son sens si la découverte initiale a disparu.

Nous voyons qu'un personnage abandonne X pour choisir Y, mais nous ne savons plus ce qu'il cherchait à accomplir, pourquoi X semblait raisonnable ni quel événement a rendu Y préférable. Remettre toutes les scènes dans l'ordre ne suffit pas. Il faut déterminer celles qui appartiennent à la même intrigue et comprendre comment chacune modifie la suivante.

L'assemblage de ces scènes forme ce que j'appelle un Épisode dans Cortex.

Un Épisode reconstruit une trajectoire de travail. Il relie une situation de départ, un objectif, les hypothèses disponibles, les décisions prises, les actions, leurs effets et le moment où la compréhension a changé. Il conserve aussi les questions restées ouvertes. S'il manque le résultat d'une action, l'Épisode reste incomplet. Si des éléments se contredisent, il ne devient pas propre par magie : la contradiction reste visible.

> Une hypothèse ne prend pas la place d'une trace absente.

Cette reconstruction ne doit pas fabriquer un récit plus cohérent que les sources. Chaque élément important reste relié aux échanges, événements ou résultats qui le fondent. Une lacune peut être nommée. Une hypothèse peut être proposée.

Un Épisode peut apporter une réponse locale. Plusieurs Épisodes composent une saison, pour rester dans l'image, et construisent une compréhension plus large. La saison suivante peut encore révéler qu'une ancienne interprétation était incomplète. L'histoire grandit sans exiger l'effacement de ses versions précédentes.

Cette unité est plus proche de l'expérience que le résumé de conversation, car elle conserve ce que l'agent cherchait à faire et ce qui s'est réellement produit. Elle reste toutefois une reconstruction. Ses frontières peuvent être discutées, ses sources peuvent être incomplètes et son résultat peut dépendre d'un facteur encore inconnu.

L'Épisode fournit ainsi une expérience exploitable, mais il ne dit pas encore ce qu'il faut en apprendre.

05

Une expérience peut produire plusieurs leçons

Imaginez un avion. Un voyant d'avarie s'allume dans le cockpit. Le commandant de bord possède une information, mais pas encore une conclusion.

Est-ce une alerte isolée ? Une panne confirmée ? Une anomalie connue du capteur ? La conséquence d'une autre avarie ? Le signal visible reste le même. Pourtant, selon la manière dont il est interprété, la liste de vérification à consulter, les contrôles à effectuer et les actions autorisées ne seront pas identiques.

Si une simple alerte devient immédiatement la preuve d'une panne, l'équipage peut appliquer la mauvaise procédure. Si une panne confirmée reste classée comme un incident mineur, la bonne procédure risque de ne jamais être déclenchée.

> La liste de vérification ne change pas l'information. Elle encadre le pouvoir qu'on lui donne.

J'ai rencontré le même problème avec les Épisodes. Un échec peut produire plusieurs interprétations : « ne jamais utiliser X », « X échoue lorsque Y manque », « l'outil était indisponible ce jour-là », « l'objectif avait été mal compris » ou « nous ne savons pas encore ce qui a provoqué l'échec ».

Ces phrases ne décrivent pas la même chose. La première propose une règle générale. La deuxième introduit une condition. La troisième rapporte une observation locale. La quatrième corrige peut-être la définition du problème. La dernière conserve une incertitude.

Si tout est stocké sous l'étiquette « apprentissage », un incident unique peut devenir une interdiction générale. L'agent rencontre une difficulté une fois et finit par éviter la situation partout.

C'est ainsi qu'un agent devient superstitieux.

Le classificateur intervient à cet endroit. Il ne décide pas si une conclusion est vraie ; il détermine ce que chaque candidat prétend être.

Dans Cortex, un candidat peut décrire une observation, une préférence, une procédure, une exception ou une croyance causale. Il peut donc rapporter un événement, exprimer un choix, proposer une manière d'agir, limiter une règle ou prétendre expliquer un résultat.

Le type change le niveau de preuve attendu. L'utilisateur est la meilleure source pour dire qu'il préfère une réponse courte, mais sa parole ne suffit pas à prouver qu'un composant technique cause une panne. Une procédure non évaluée ou une exception observée une fois

doit conserver cette limite.

Une étude sur le comportement des agents face à leur mémoire montre qu'ils tendent à reproduire une réponse lorsque la situation présente ressemble à une expérience rappelée. Une erreur ou une expérience mal alignée peut ainsi se propager aux tâches suivantes [7].

Le classement évite de traiter toutes les interprétations de la même manière. Un candidat correctement classé peut encore être faux, trop général ou soutenu par plusieurs sources issues du même événement.

Classer n'est pas valider. Il faut ensuite décider quelle matière autorise un apprentissage à agir.

06

Donner du pouvoir à un apprentissage

Un apprentissage candidat peut être très convaincant. Il peut être bien formulé, cohérent avec les événements disponibles et produit par un modèle performant. Rien de cela ne constitue encore une preuve indépendante.

Reprenons l'avion. Après une avarie, le compte rendu de l'équipage peut proposer une explication plausible. Elle ne réécrit pas pour autant les listes de vérification de tous les appareils. Les données du vol, la maintenance et des événements comparables doivent encore la confirmer, la limiter ou la contredire. En attendant, l'hypothèse reste sous surveillance.

C'est le rôle de la quarantaine dans la vision de Cortex. Lorsqu'un worker extrait un apprentissage possible d'un Épisode, il ne l'active pas dans le même mouvement. Il conserve le candidat, ses sources, son périmètre et ses limites. Une politique distincte décide ensuite si la matière disponible suffit pour l'autoriser à intervenir dans une mission.

Toutes les affirmations ne demandent pas les mêmes garanties. Une personne peut définir seule sa préférence pour le tutoiement. Si elle affirme qu'un outil provoque toujours une panne, sa parole reste une source, mais elle ne transforme pas une association en cause générale.

Il faut alors séparer la confiance de la vraisemblance. Un modèle peut trouver une conclusion crédible parce qu'elle s'insère bien dans le récit. La confiance du système répond à une question plus froide : quelle matière soutient cette conclusion ?

> Une source répétée dix fois ne devient pas dix preuves.

Deux sessions peuvent reprendre la même note et plusieurs résumés dériver du même événement. Sans suivi de leur origine, la répétition produit seulement une illusion de confirmation.

La provenance permet de remonter de l'apprentissage à l'Épisode, puis aux traces, en distinguant ce qui soutient, contredit, limite ou invalide. Si cette chaîne se casse, la confiance doit être revue plutôt que continuer comme si la matière était intacte.

Cette politique traduit un niveau de risque accepté. Une préférence locale et réversible peut recevoir du pouvoir rapidement. Une croyance causale capable de modifier une mission ou de libérer une action demande une matière plus solide. Plus la conséquence possible est importante, plus cette matière doit être explicite.

Dans Cortex, un candidat peut commencer en quarantaine, devenir actif, puis être contesté ou déclaré obsolète. Ces états ne disent pas seulement comment l'affirmation est considérée ; ils déterminent ses droits. En quarantaine, elle reste visible sans gouverner une mission. Active, elle peut entrer dans un souvenir dans son périmètre d'admission. Contestée, elle apparaît comme une incertitude. Révoquée, elle conserve sa place dans l'histoire mais perd son pouvoir opératoire.

Le « pouvoir opératoire » désigne cette capacité à modifier le contexte actif, orienter une décision ou participer à l'autorisation d'une action. Deux éléments également consultables peuvent ainsi posséder des droits d'usage différents. La mémoire cesse d'être un entrepôt uniforme : chaque élément porte une origine, une portée, un statut et un droit d'usage.

Des recherches récentes étudient la même difficulté sous plusieurs angles : réviser une croyance lorsque la situation change, régler la dépendance d'un agent à ses souvenirs ou empêcher une mémoire issue d'une source non fiable de justifier une action sensible [8][9]. Cortex prolonge cette logique jusqu'à l'usage : un apprentissage devient utile lorsque le système sait aussi limiter le pouvoir qu'il lui accorde.

07

Conserver une erreur sans la laisser agir

Tôt ou tard, une nouvelle preuve contredit ce que le système avait appris.

Supprimer l'ancienne croyance évite son retour, mais détruit ce qui l'avait rendue plausible. Nous perdons alors la possibilité de reconnaître plus tard une situation capable de produire la même illusion.

Tout conserver avec la mention « ne pas utiliser » préserve l'histoire, mais la garantie reste faible. Si le texte revient dans le contexte, le modèle peut encore le mélanger à la correction ou reproduire une partie de l'ancien raisonnement.

Il faut séparer la conservation et l'autorisation.

Imaginons qu'une ancienne liste de vérification repose sur une mauvaise interprétation. Elle ne doit plus guider l'équipage, mais sa disparition effacerait les incidents qui semblaient la confirmer et la preuve qui a montré son défaut. Elle reste donc dans les archives sans figurer parmi les procédures autorisées en vol.

> Révoquer ne signifie pas effacer.

J'emploie le mot « révocation » pour désigner ce retrait du pouvoir opératoire. La croyance reste reliée aux éléments qui l'avaient soutenue et à ceux qui l'ont remise en cause, mais son nouveau statut lui retire la capacité de gouverner une situation future.

Cette distinction paraît sémantique jusqu'au moment où l'agent peut agir. Écrire dans un prompt « n'utilise pas les informations invalides » revient à confier la règle au modèle. Or le modèle est précisément l'élément que la règle doit encadrer. Une garantie qui dépend de sa bonne volonté n'est pas une séparation technique.

Le contrôle doit donc intervenir avant l'injection dans le contexte et être vérifié de nouveau au moment de l'usage. Si un apprentissage a été révoqué entre la préparation du souvenir et la proposition d'action, le système doit pouvoir l'écarter à ce second passage.

Le refus doit aussi être attribuable. Écarter silencieusement un élément protège peut-être la décision, mais empêche de comprendre ce qui s'est passé. Cortex nomme donc l'artefact écarté, son statut et la raison du refus, sans bloquer les autres éléments admissibles. Une croyance problématique ne fait pas disparaître tout le souvenir.

Cette propriété répond à une question formulée très clairement lors d'un échange public autour de Cortex : pour une décision précise, le système peut-il attester que seuls des artefacts admissibles ont franchi la frontière décisionnelle ?

Je ne cherche pas à prouver qu'une ancienne croyance n'exerce plus aucune influence latente dans un modèle opaque. Cette preuve serait hors de portée. Je travaille sur le chemin observable : les éléments récupérés, leurs versions, leurs statuts, la politique appliquée, les refus, la proposition et l'action éventuellement libérée.

Cortex relie cette frontière au cycle qui va des traces de travail jusqu'à l'action. Les mémoires versionnées, les lignées de provenance et les barrières de contrôle des actions sensibles apportent ici des points de comparaison directs [6][9][10].

La révocation traite le pouvoir d'une ancienne croyance. Elle ne résout pas la sélection : parmi des milliers d'expériences, lesquelles concernent la situation présente ?

08

Retrouver n'est pas encore se souvenir

Une recherche classique s'appuie beaucoup sur les mots. Elle fonctionne bien pour un nom, une référence ou une expression précise. Elle devient moins fiable lorsque la même idée revient sous une formulation différente.

Dans mon désaccord sur la conversation vieille de trois semaines, je me souvenais d'un « impact ». L'autre personne parlait de « tout casser ». Une recherche sur le premier terme peut manquer la seconde formulation. Une recherche sur le second peut ramener des documents qui parlent de casse sans rapport avec notre sujet.

Cortex utilise notamment BGE-M3 pour rapprocher les textes par leur sens. C'est un modèle multilingue qui transforme un texte en une empreinte numérique afin de comparer sa proximité avec d'autres textes [11]. Le terme technique est *embedding*. Dans Cortex, cette empreinte dense comporte 1 024 valeurs. Le nombre importe moins que la fonction : donner des coordonnées à une formulation pour retrouver des voisines.

Imaginez un immense nuancier. Du jaune au vert, puis du vert au bleu, il existe une continuité de teintes. Certaines sont plus proches du jaune, d'autres du vert ou du bleu. Deux phrases peuvent occuper des zones voisines même si elles n'emploient pas les mêmes mots.

« Cet élément aura un impact important » et « cet élément risque de provoquer une rupture » ne sont pas équivalents. Ils partagent néanmoins une partie de leur sens. La recherche sémantique peut les rapprocher et les proposer comme candidats.

Le mot « empreinte » aide aussi à comprendre la limite. Une empreinte de patte de chat ressemble davantage à celle d'un lion, d'un tigre ou d'une panthère qu'à celle d'un cerf. Elle indique une famille possible. Elle ne prouve pas quel animal est passé. Il faut encore examiner la taille, le terrain et les autres traces.

> **La proximité sémantique sait ce qui ressemble. Elle ne sait pas, à elle seule, ce qui est vrai, actuel, admissible ou utile.**

Un ancien apprentissage révoqué peut être très proche de la nouvelle situation. Une règle issue d'un autre projet peut employer exactement les bons mots. Une expérience récente peut sembler pertinente alors que sa source n'est plus disponible. Si la recherche s'arrête au classement par ressemblance, elle remet du pouvoir entre les mains du score.

Le rappel doit donc combiner plusieurs signaux. Les mots exacts restent utiles pour les noms, les identifiants et les formulations précises. La proximité de sens élargit la recherche. Les relations du graphe apportent les sources, les contradictions et les liens entre expériences. La politique d'admissibilité retire ce qui n'a pas le droit de revenir dans le contexte actif.

L'ordre compte. La recherche propose des candidats ; elle ne décide pas seule du souvenir. Même filtrée, la matière retrouvée peut encore être trop abondante pour la mission.

09

Le souvenir comme pack construit pour une situation

Avant un vol, le commandant de bord ne reçoit pas toutes les archives de la compagnie, tous les dossiers de maintenance et chaque compte rendu écrit depuis vingt ans. Il reçoit un briefing préparé pour ce vol : la destination, la météo, l'état de l'appareil, les contraintes connues et les procédures susceptibles de devenir utiles.

Un autre vol produira un autre briefing à partir des mêmes archives.

Pourquoi ne pas tout transmettre ? Parce qu'une information peut être pertinente sans être utile maintenant. Accumuler les documents finit par cacher les éléments importants au milieu du bruit. Un modèle rencontre la même limite. Son contexte possède une capacité finie, et chaque souvenir injecté peut modifier son raisonnement.

Le souvenir prend ici la forme d'un pack contextuel. Le pack part d'une situation précise : le message actuel, l'objectif, le sujet concerné, l'état de la mission, ses contraintes et la place disponible.

Le système sélectionne ensuite les apprentissages admissibles et rattache les Épisodes nécessaires pour les comprendre. Il retire les doublons, conserve les sources et rend visibles les contradictions ou les limites qui accompagnent une affirmation. Un apprentissage actif et directement lié à la mission passe avant une expérience ancienne simplement voisine.

Le budget compte. Si dix éléments sont pertinents mais que la place disponible n'en accepte que cinq, la sélection doit l'indiquer. Dépasser silencieusement la capacité du modèle ne rend pas le souvenir plus complet ; cela déplace le problème vers le bruit et le coût.

> **Le pack n'est pas la mémoire entière. Il s'agit de la partie autorisée à prendre place sur le bureau pour cette mission.**

Ce rattachement à une situation protège aussi contre la réutilisation automatique. Le pack porte le tour qui l'a produit, son périmètre et les versions mobilisées. Une modification de l'objectif ou du sujet peut exiger un nouveau pack. Une sélection adaptée à un instant ne

devient pas une vérité universelle.

Le système peut enfin conserver trois catégories qui se confondent facilement : les éléments présentés, ceux que l'agent a voulu mobiliser et ceux qui ont été refusés. Voir une information dans le contexte ne prouve pas qu'elle a causé la décision. Ne pas enregistrer son passage empêcherait cependant toute enquête ultérieure.

Le pack arrive ensuite à la frontière la plus sensible : celle où une proposition peut devenir une action.

10

Attester la frontière décisionnelle

Comment savoir si un souvenir a réellement changé une décision ?

Nous pourrions demander au modèle pourquoi il a agi. Sa réponse resterait une interprétation produite après coup. Elle peut être utile, mais elle ne remplace pas l'enregistrement du chemin emprunté.

Revenons une dernière fois à l'avion. Une boîte noire ne lit pas les pensées du commandant et n'enregistre pas tout. Il reste des événements incomplets et des décisions dont la cause exacte n'apparaît nulle part. Les enquêteurs formulent alors des hypothèses, mais ils ne remplissent pas les espaces au hasard. Chaque hypothèse doit rester compatible avec les traces et indiquer ce qui permettrait de la confirmer ou de l'écarter.

> Une hypothèse ouvre une piste. Elle ne réécrit pas les données manquantes.

Cortex cherche à conserver une chaîne comparable pour ses décisions instrumentées. Lorsqu'un pack est présenté, le système garde son identité, le tour concerné et les apprentissages qu'il contient. Si certains sont mobilisés, leur version, leur statut et la raison donnée pour cet usage sont enregistrés. Les refus restent associés au même passage.

Cette provenance accompagne ensuite la proposition d'action. Elle peut être reliée à l'autorisation, à l'opération exécutée et au résultat connu. Nous obtenons un chemin observable : la situation, le souvenir présenté, les éléments retenus ou refusés, la proposition, l'autorisation, l'action et son effet.

Dans Cortex, cette photographie prend la forme d'une attestation de mémoire, dérivée de l'état durable du système plutôt que rédigée par le modèle. Elle associe à une proposition les artefacts qui ont été admis ou refusés au passage de la frontière.

La conséquence est concrète : une action automatisée ne doit pas être libérée lorsque le système ne peut pas établir la provenance cognitive attendue. La proposition peut rester visible et être confirmée par une personne, mais l'automatisation s'arrête.

Cette règle ne dit pas que l'action est fausse. Elle dit que le système ne possède pas la preuve attendue pour l'exécuter sans intervention humaine.

MemLineage et un brouillon IETF publié en 2026 travaillent également ce lien entre provenance, autorisation et action [9][10]. Ils donnent un point de comparaison à la frontière que Cortex matérialise dans son propre cycle.

L'attestation documente un passage observable. Elle ne mesure pas à elle seule combien chaque souvenir a causé la décision. Une réussite ne confirme donc pas automatiquement l'apprentissage, pas plus qu'un échec ne suffit à le révoquer. Le résultat devient une nouvelle pièce du dossier et doit revenir vers l'Épisode qui avait produit la décision.

11

Le Rêve : reprendre les expériences lorsque l'urgence est passée

Pendant une mission, le système doit répondre, utiliser des outils et parfois demander une autorisation. Ce n'est pas toujours le bon moment pour comparer des dizaines d'Épisodes, rechercher des contradictions anciennes ou réévaluer une confiance.

J'ai appelé « Rêve » le second temps consacré à cette consolidation. Le nom vient de la découverte et de la communication. Il ne signifie pas que Cortex imagine librement ou reproduit le sommeil humain.

Le Rêve est un traitement à froid. Il reprend les Épisodes lorsque la décision immédiate ne presse plus. Il peut comparer des situations, rapprocher des résultats, chercher plusieurs expériences indépendantes et signaler qu'une ancienne confiance ne correspond plus à la matière disponible.

Plusieurs expériences concordantes peuvent renforcer un apprentissage. Une contradiction peut le rendre à nouveau contesté. Une hypothèse ne doit jamais profiter de ce traitement pour se promouvoir seule. Les recommandations produites par un modèle repassent par les mêmes règles de validation et de pouvoir opératoire.

> Le Rêve ne réécrit pas le passé. Il modifie ce que le système est autorisé à en conclure.

La consolidation différée existe dans d'autres architectures de mémoire, parfois sous le nom de « Dream State » [3][6]. Dans Cortex, sa place est celle du retour d'expérience : reprendre les conséquences d'une action, les rattacher à l'Épisode et réévaluer les apprentissages sans perdre leurs sources.

Ce retour sépare une mémoire gouvernée d'une expérience cumulative. Tant que les conséquences ne modifient pas l'interprétation, le système peut rappeler ce qu'il croyait savoir, mais il n'apprend pas complètement de ce que son action a produit.

La chaîne suit alors le travail de bout en bout : capter, reconstruire un Épisode, extraire des apprentissages candidats, leur donner un statut, retrouver les éléments utiles, construire un pack, attester leur passage, observer le résultat, puis ramener ce résultat vers l'expérience. Cette dernière liaison reste celle que Cortex doit fermer en usage réel.

Conclusion — Du souvenir à la responsabilité

Au début de ce travail, je demandais si un agent pouvait apprendre alors que son modèle restait figé.

Ma réponse reste oui, à condition de situer l'apprentissage au bon endroit. Le modèle n'a pas besoin de modifier ses paramètres après chaque mission. Le système qui l'entoure peut conserver des traces, reconstruire des expériences et modifier durablement la matière présentée lors des situations suivantes.

Dans ce texte, apprendre signifie davantage qu'accumuler des informations. Le système transforme ses traces en Épisodes sourcés, puis en apprentissages révisables. Il rend cette expérience utilisable à travers une politique capable de limiter son pouvoir : quarantaine, admission, contestation, révocation, rappel borné et refus attribuable.

Le souvenir n'est alors ni un fichier ni une copie fidèle du passé. C'est une reconstruction produite pour une situation présente à partir d'artefacts admissibles. Il conserve assez d'histoire pour expliquer pourquoi une ancienne croyance avait paru raisonnable, sans lui rendre automatiquement le droit de gouverner une nouvelle action.

L'ambition est de rendre la faillibilité de cette mémoire visible, traçable et corrigeable. Les frontières du système rendent observable ce qui a été proposé, admis, refusé, présenté et autorisé.

Cortex est l'implémentation dans laquelle j'explore cette chaîne. Il assemble mémoire épisodique, relations temporelles, révision de croyances, recherche sémantique, provenance et contrôle d'actions autour d'une question centrale : quel pouvoir une expérience passée reçoit-elle sur la décision présente ?

Ce travail sur le souvenir devait néanmoins venir avant une autre question.

Avant Cortex, j'avais tenté de construire ENOD, un Agent à Homéostasie Déontologique. Je cherchais à isoler certains garde-fous que nous associons à la peur, à la honte, à la culpabilité et à la responsabilité, puis à simuler mécaniquement leurs conséquences utiles. Il ne s'agissait pas de faire ressentir des émotions à une machine. La peur devenait une anticipation du risque, la honte une invalidation globale d'un résultat incompatible, la culpabilité une dette de réparation et la responsabilité un verrou sur la libération d'une action.

Cette expérience s'est arrêtée. Pour qu'une dette survive à une session, qu'un incident passé modifie une décision future ou qu'une réparation reste due, il fallait d'abord une mémoire capable de porter cette continuité.

> J'avais cherché les garde-fous avant d'avoir construit la matière qui devait les nourrir.

Le travail sur le souvenir fournit maintenant cette matière. Il ne répond pas encore à la question suivante, mais il permet enfin de la poser correctement : des expériences passées peuvent-elles créer et nourrir une responsabilité artificielle ?

Annexes de lecture — La matérialisation actuelle dans Cortex

Annexe A — L'attestation de mémoire

L'attestation distingue aujourd'hui quatre situations : la provenance n'a pas été calculée, elle a été calculée sans mémoire mobilisée, des artefacts précis ont été mobilisés, ou la provenance n'a pas pu être lue. Ces états empêchent de confondre « rien n'a été utilisé » avec « le système ne sait pas ce qui a été utilisé ».

Lorsqu'une mémoire intervient, l'attestation conserve les identifiants, les versions, les statuts et les formulations nécessaires. Une modification ou une purge ultérieure ne retire donc pas à la proposition le fondement qui existait au moment de sa création. Une empreinte permet de détecter une modification de cette photographie dans le journal interne.

Cette matérialisation agit déjà sur l'autorisation. Une permission permanente ne libère pas automatiquement l'action lorsque la provenance cognitive n'est pas établie. La proposition reste ouverte à une confirmation humaine. L'attestation demeure toutefois une preuve interne au système ; elle n'offre pas encore le niveau de vérification portable ou cryptographique proposé par certains travaux voisins [9][10].

Annexe B — L'état de la boucle du Rêve

Cortex sait construire des Épisodes, classifier des candidats, préparer un rappel, enregistrer des refus et attester la mémoire associée à une proposition d'action. Le résultat d'une action peut également être conservé.

Le raccord systématique reste à fermer : transformer ce résultat en nouvelle matière de l'Épisode, puis déclencher la réévaluation complète des apprentissages concernés. Tant que cet effet n'est pas démontré dans l'usage réel, le Rêve décrit une chaîne partiellement matérialisée et un programme de consolidation à achever.

Glossaire de lecture

Agent : système complet formé par un modèle, un environnement d'exécution, des outils, des règles, des permissions et une mémoire.

Modèle : moteur de génération ou de raisonnement sollicité par l'agent.

Trace : enregistrement brut d'un événement, d'un échange, d'un appel d'outil ou d'un résultat.

Interprétation : lecture proposée à partir d'une ou plusieurs traces.

Épisode : reconstruction bornée d'une trajectoire de travail, avec situation, objectif, décisions, actions, résultat et sources.

Apprentissage candidat : affirmation dérivée d'une expérience qui n'a pas encore reçu de pouvoir opératoire.

Pouvoir opératoire : capacité accordée à un artefact mémoire pour entrer dans le contexte directif, orienter une décision ou participer à l'autorisation d'une action.

Quarantaine : état dans lequel un candidat reste conservé et analysable sans pouvoir gouverner une mission.

Révocation : retrait du pouvoir opératoire sans suppression de l'histoire ni des sources.

Rappel : recherche de matière potentiellement utile à une situation.

Pack contextuel : sélection bornée d'artefacts admissibles préparée pour un tour et une situation précis.

Attestation : photographie dérivée de l'état durable qui conserve ce qui a franchi ou non une frontière instrumentée.

Rêve : consolidation différée et réévaluation à froid des expériences et des apprentissages.

Limites et programme de validation

La vision devient testable à partir du moment où ses effets attendus sont nommés. Le programme de validation devra établir :

- si les missions s'améliorent durablement par rapport à une mémoire fondée sur un historique, un résumé ou un rappel vectoriel seul ;
- si une croyance révoquée reste consultable tout en cessant de franchir les frontières directives et automatiques instrumentées ;
- si les règles d'admission écartent les généralisations injustifiées sans bloquer trop de souvenirs utiles ;
- quelle influence un artefact exerce réellement sur une décision, en adaptant au contexte décisionnel le principe *leave-one-resource-out* utilisé par ProvenAI sur des réponses générées [12] ;
- si le résultat d'une action revient effectivement vers l'Épisode, modifie les apprentissages concernés et change une mission ultérieure ;

- quel coût cette gouvernance ajoute en temps, en calcul et en volume de contexte.

Ces expériences permettront de mesurer la portée de la proposition, de la comparer aux architectures proches et de corriger les mécanismes qui ne produisent pas l'effet attendu.

Sources et prolongements

Les références suivantes situent les mécanismes décrits et soutiennent les principaux constats de recherche mobilisés dans le texte.

- [1] Noah Shinn et al., *Reflexion: Language Agents with Verbal Reinforcement Learning*, 2023.
- [2] Andrew Zhao et al., *ExpeL: LLM Agents Are Experiential Learners*, 2023.
- [3] Jiaquan Zhang et al., *Lightweight LLM Agent Memory with Small Language Models*, ACL 2026.
- [4] Preston Rasmussen et al., *Zep: A Temporal Knowledge Graph Architecture for Agent Memory*, 2025, prépublication.
- [5] Pratyay Banerjee et al., *APEX-MEM: Agentic Semi-Structured Memory with Temporal Reasoning for Long-Term Conversational AI*, ACL 2026.
- [6] Kumiho Inc., *Graph-Native Cognitive Memory for AI Agents: Formal Belief Revision Semantics for Versioned Memory Architectures*, 2026, prépublication d'entreprise.
- [7] Zidi Xiong et al., *How Memory Management Impacts LLM Agents: An Empirical Study of Experience-Following Behavior*, ACL 2026.
- [8] Hanxiang Chao et al., *STALE: Can LLM Agents Know When Their Memories Are No Longer Valid?*, 2026, prépublication.
- [9] Ciyan Ouyang et Rui Hou, *MemLineage: Lineage-Guided Enforcement for LLM Agent Memory*, 2026, prépublication.
- [10] K. Rampalli, *Binding Per-Action Authorization and Memory Provenance into Agent Action Capsules*, Internet-Draft, juillet 2026, travail en cours.
- [11] Jianlv Chen et al., *BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation*, 2024.
- [12] Mohammad Faizan et Dalal Alharthi, *ProvenAI: Provenance-Native Traces of Evidence in Generated Answers*, 2026, prépublication.
- [13] SurrealDB, *What is SurrealDB* et *Graph model*, documentation officielle consultée le 5 août 2026.