CONTROL

De l'autorisation à l'effet

Comment Sawbill gouverne l'exécution des agents

Comment Sawbill place une frontière de contrôle entre ce qu'un agent propose et ce qui peut modifier un système réel.

LECTEURS

Architectes de plateforme / Responsables sécurité / Responsables DevOps et SRE / Responsables de plateforme IA

Sommaire

De l'autorisation à l'effet

L'enjeu en une minute

Un agent utile doit pouvoir faire plus que produire du texte. Il doit lire un dépôt, exécuter des outils, lancer des validations et parfois demander une mutation Git, cloud ou applicative. Lui remettre directement les credentials nécessaires revient toutefois à confondre sa capacité cognitive avec un droit d'action.

Sawbill introduit une séparation structurante :

> **l'agent propose et produit ; un chemin d'admission distinct décide ; un point d'application contrôlé détient le credential et réalise l'effet.**

Cette séparation permet d'augmenter l'autonomie sans transformer chaque agent en administrateur permanent.

L'autonomie n'est pas l'autorité

Dans une chaîne agentique classique, un orchestrateur appelle un modèle, lui transmet des outils et considère la tâche terminée lorsque le fournisseur renvoie `Completed`. Cette approche laisse plusieurs questions sans réponse :

- le paquet de travail exécuté correspond-il encore à la demande autorisée ?
- les politiques, branches, budgets et identités sont-ils toujours valides ?
- le worker peut-il appeler directement un fournisseur externe ?
- un timeout signifie-t-il échec, succès ou résultat inconnu ?
- une reprise relance-t-elle sans le savoir un effet déjà produit ?

Sawbill ne traite pas l'exécution comme une simple file de jobs. Il la traite comme une suite de transitions gouvernées, chacune avec son propriétaire et son plafond de garantie.

Le parcours d'exécution Sawbill

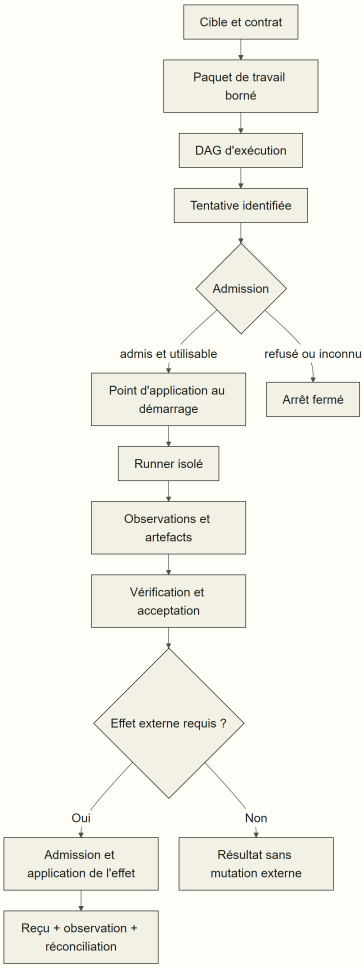

Figure 1 - Sawbill

Ce chemin évite deux extrêmes : un agent sans aucune capacité pratique, ou un agent doté de credentials généraux et surveillé seulement après coup.

1. Transformer l'intention en travail borné

Le paquet de travail décrit ce qu'une exécution peut tenter. Il lie notamment :

- un objectif et des critères de sortie ;
- les entrées exactes ;
- le périmètre de lecture et d'écriture ;
- les outils ou capacités admissibles ;
- les politiques et profils de vérification ;
- les limites de temps, de coût et de données ;
- les effets explicitement permis ou interdits.

Le paquet n'est pas un prompt enrichi. Il constitue une frontière entre la mission gouvernée et les choix cognitifs laissés à l'agent.

Un DAG d'exécution ordonne ensuite les paquets selon des dépendances explicites. Ce DAG ne remplace ni l'architecture du système ni le graphe de ses exigences. Il répond seulement à la question : dans quel ordre ce travail peut-il être tenté ?

2. Une tentative est une invocation, pas une tâche éternelle

Une tentative possède une identité propre. Elle correspond à une invocation physique sous un ensemble précis d'entrées, de politiques et de capacités.

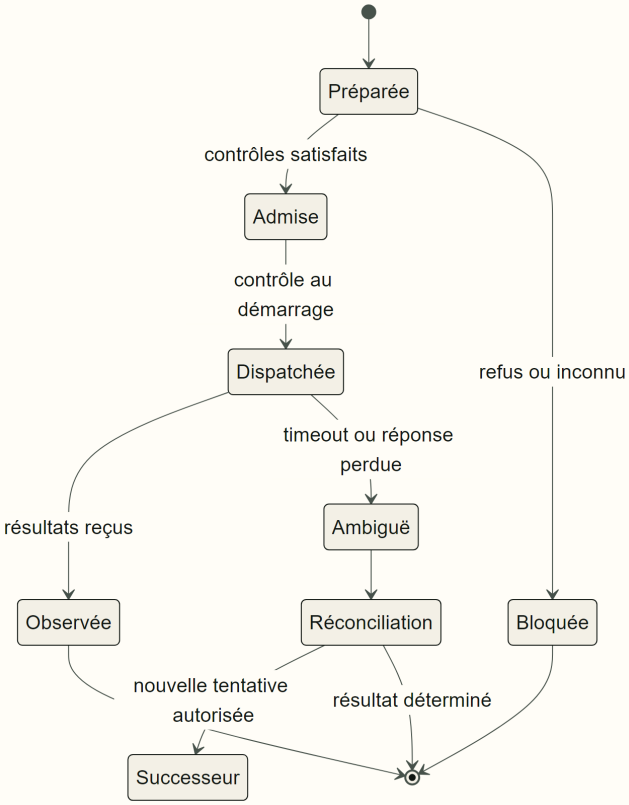

Figure 2 - Sawbill

Une reprise ne réécrit pas la tentative précédente. Un retry, un changement de fournisseur ou un redémarrage crée un successeur. Le nouveau contexte ne récupère pas automatiquement les anciennes autorisations, budgets ou capacités.

Cette règle conserve l'histoire et empêche qu'une relance technique devienne un rejeu silencieux d'autorité.

3. L'admission décide ; le point d'application contrôle

Sawbill distingue la décision de politique de son application physique :

- le moteur d'admission évalue les entrées exactes et produit une décision historique ;
- le point d'application de politique, ou PEP, relit l'état courant avant le démarrage ;
- le runner exécute uniquement la capacité qui lui est confiée ;
- le coordinateur organise la procédure sans posséder un second chemin d'autorité.

Cette architecture suit l'esprit du zero trust : aucune confiance implicite n'est accordée parce que le worker tourne sur le bon réseau, dans le bon cluster ou sous le bon compte de service. Le NIST SP 800-207 distingue déjà authentification et autorisation avant l'accès à une ressource ; Sawbill étend cette discipline aux tentatives agentiques et à leurs objets exacts.

Au moment du dispatch, le PEP peut vérifier :

- l'identité et la qualification du workload ;
- la version du paquet et de ses entrées ;
- la validité actuelle des politiques et autorisations ;
- la branche, le head ou l'état de ressource attendu ;
- le budget et les capacités encore utilisables ;
- l'absence d'invalidation depuis la décision initiale.

Une décision historiquement positive ne suffit donc pas si le contexte a changé.

4. Intégrer Sawbill sans remplacer la chaîne de livraison

Sawbill s'insère entre les outils qui proposent ou exécutent du travail et les ressources dont la mutation engage l'organisation. Les orchestrateurs, runtimes, CI, runners et plateformes cloud peuvent rester en place. Leur rôle change : ils reçoivent une mission bornée et renvoient des observations ; ils ne portent plus à eux seuls l'autorité, l'acceptation ou le droit d'effet.

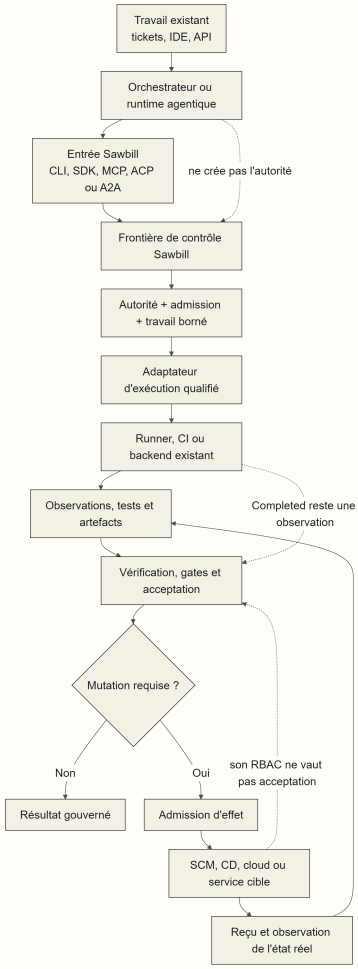

Figure 3 - Sawbill

Le déplacement de frontière est précis : avant Sawbill, l'orchestrateur possède souvent le prompt, le contexte, les outils et les credentials d'effet. Avec Sawbill, il conserve le flow cognitif, tandis que l'autorité, l'admission, la preuve et l'effet passent par des contrôles distincts et vérifiables.

Familles d'intégration

Point de la chaîne	Interfaces et exemples d'extrémités	Rôle conservé par Sawbill
Gestion du travail	API et webhooks de Linear, Plane, GitHub, GitLab ou Azure DevOps	Le ticket reste une projection ; il ne devient ni contrat ni verdict
Runtimes agentiques	CLI, SDK, MCP, ACP, A2A ; runtimes tels qu'OpenCode ou Microsoft Agent Framework	L'adaptateur transporte une exécution bornée ; le runtime ne s'auto-admet pas
Orchestration et calcul	Backends de workflow, CI, conteneurs, Kubernetes ou batch	Le statut externe reste une observation à corréler et vérifier
SCM et livraison	Dépôts, checks, pipelines et systèmes de déploiement	Fusion, release et déploiement restent des effets séparément admis
Identité, secrets et clés	IdP, identité de workload, coffre de secrets, KMS ou HSM	Les credentials restent derrière leur frontière ; ils ne deviennent pas autorité métier
Observabilité	Logs, traces, audit logs et bus d'événements	Les signaux sont attribués, expurgés et liés avant de devenir des preuves utilisables

Ces noms illustrent des extrémités d'intégration documentées ; ils ne constituent ni une dépendance architecturale, ni une matrice de disponibilité par version. La compatibilité effective dépend toujours du profil et de la qualification de l'adaptateur retenu.

5. L'agent ne détient pas le credential effecteur

Le contrôle le plus important intervient lorsque le travail doit produire une mutation externe : pousser un commit, fusionner une branche, modifier une ressource cloud, envoyer un message ou déclencher un déploiement.

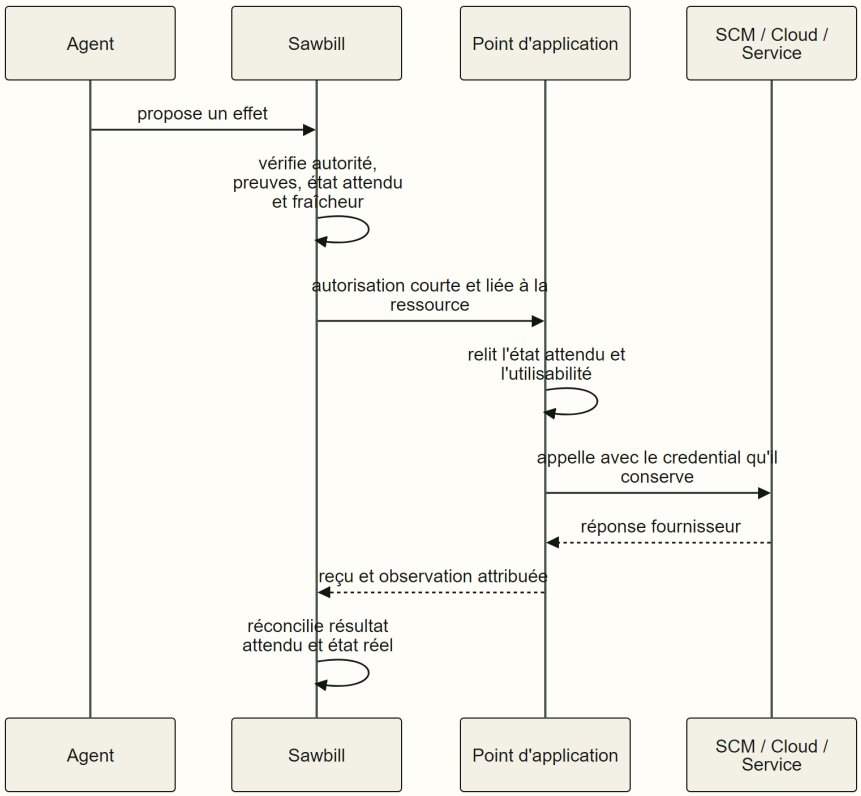

Figure 4 - Sawbill

Le runner ne reçoit pas le secret et ne peut pas contourner le point d'application. La capacité d'effet est courte, liée à une finalité, une ressource et un état attendu. Elle ne devient pas un bearer token général.

Cette séparation réduit le rayon d'explosion d'un prompt hostile, d'un outil compromis ou d'un agent qui sort de son périmètre.

6. Un effet possède sa propre admission

L'autorisation d'exécuter un agent ne vaut pas autorisation de modifier une ressource. Sawbill sépare donc :

1. l'admission de la tentative ;
2. la production et la vérification de ses résultats ;
3. l'admission de l'effet externe ;
4. l'application de cet effet par le point de contrôle de la ressource.

Un agent peut ainsi compiler, tester ou préparer un changement sans obtenir le droit de le publier. De même, un résultat accepté peut nécessiter une nouvelle décision avant un déploiement.

Le contrôle porte sur l'état attendu. Une demande « modifier la branche si son head vaut X » est refusée si le head est devenu Y. Cette vérification protège contre les décisions valides prises sur un monde qui n'existe plus.

7. Le succès du fournisseur reste une observation

Un code HTTP 2xx signifie qu'une requête a été reçue ou traitée selon la sémantique du service ; il ne prouve pas que l'objectif métier est atteint. La RFC 9110 fournit la sémantique du protocole HTTP, pas celle de l'acceptation Sawbill.

Sawbill conserve donc séparément :

- la persistance de la demande ;
- l'acceptation du démarrage par le fournisseur ;
- l'observation d'un processus ;
- la réponse du service ;
- l'état réel observé après l'effet ;
- la décision d'assurance ou d'acceptation.

Un statut `Completed` provenant d'un moteur de workflow, d'un CI ou d'un agent runtime ne devient pas automatiquement une gate positive.

8. L'ambiguïté déclenche une réconciliation, pas un replay

Les systèmes distants produisent des résultats ambigus : la requête a pu être appliquée alors que la réponse a été perdue. Relancer immédiatement est dangereux, surtout pour une mutation non idempotente.

Sawbill ouvre alors un cas de réconciliation :

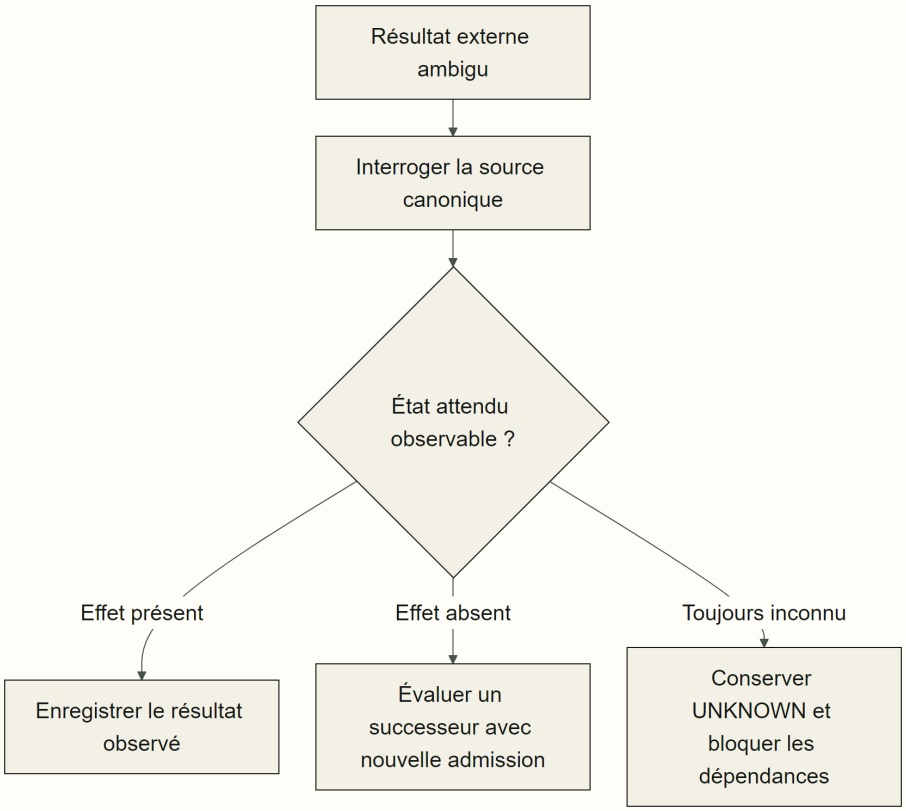

Figure 5 - Sawbill

L'incertitude n'est ni un succès optimiste ni un échec autorisant automatiquement un retry. Elle reste visible jusqu'à ce qu'une observation qualifiée permette de continuer.

9. Paralléliser sans additionner des permissions individuelles

Deux paquets admissibles séparément ne le sont pas nécessairement ensemble. Ils peuvent partager une ressource, dépasser un budget agrégé ou produire des effets non commutatifs.

Sawbill évalue donc un ensemble parallèle comme un ensemble : dépendances, conflits, capacités, budgets et effets sont vérifiés sur un même cut. La sélection reste une proposition de scheduling, jamais une autorité supplémentaire.

Cette propriété devient essentielle lorsqu'un orchestrateur lance plusieurs agents spécialisés sur un même portefeuille de dépôts.

Scénario : préparer et publier l'export de données

Reprenons l'ajout d'une fonction d'export de données clients.

1. Le plan crée des paquets distincts pour l'API, le contrôle d'accès, les tests et la configuration de déploiement.
2. L'agent développeur reçoit uniquement les sources et outils nécessaires à son paquet.
3. L'admission vérifie la version des entrées, le budget et l'autorité déléguée.
4. Le point d'application démarre un runner isolé sans lui remettre de credential de production.
5. Le runner produit une proposition, des tests et des observations.
6. Les contrôles de qualité et de sécurité sont évalués séparément.
7. Une demande d'effet propose la fusion du commit exact sur le head attendu.

8 L'adaptateur SCM conserve le credential, relit le head et effectue la mutation.

9 Un reçu est conservé, puis la branche et le déploiement sont observés.

10 Si le fournisseur répond de manière ambiguë, la réconciliation précède toute nouvelle tentative.

La chaîne permet à l'agent de travailler vite tout en empêchant sa sortie de périmètre de devenir immédiatement un effet réel.

Valeur pour l'entreprise

Autonomie à moindre rayon d'explosion

Les agents peuvent explorer et produire sans recevoir des secrets généraux. Les effets sensibles restent concentrés dans des points d'application durcis.

Contrôle cohérent entre outils

Un runtime local, un moteur de workflow, un CI ou un backend distant reste un adaptateur d'exécution. Le changement de fournisseur ne change pas la sémantique d'autorité, d'admission ou d'acceptation.

Reprise explicable

Les tentatives, retries et effets ambigus restent distincts. Les opérations peuvent reconstruire ce qui a été demandé, observé et réconcilié sans inventer un résultat à partir d'un log partiel.

Sécurité avant l'effet

Sawbill ne se limite pas à détecter une action après coup. Il positionne les contrôles sur le chemin de la mutation.

Ce que chaque signal permet réellement de conclure

Signal	Conclusion permise	Conclusion interdite
Paquet sélectionné	Le travail est candidat au scheduling	Il peut démarrer
Admission positive	Les entrées exactes ont satisfait la politique au cutoff	Le contexte est encore frais
Dispatch accepté	Le point d'application a soumis le démarrage borné	Le travail est correct ou terminé
Fournisseur `Completed`	Le fournisseur a déclaré son état	Les gates Sawbill sont satisfaites
Reçu d'effet	Une requête précise a été observée selon le profil	L'objectif réel est atteint
État réel observé	Une condition a été constatée dans un périmètre	Le livrable entier est acceptable

Références officielles

Source	Statut	Apport utilisé	Limite conservée par Sawbill
NIST SP 800-207 - Zero Trust Architecture	Publication NIST finale	Décisions d'accès centrées sur la ressource, sans confiance implicite liée au réseau	Sawbill ajoute les bindings propres aux actes, tentatives et effets agentiques
NIST SP 800-218 - Secure Software Development Framework 1.1	Publication NIST finale	Pratiques de développement logiciel sécurisé et vocabulaire commun	Le framework ne fournit pas l'admission runtime Sawbill
RFC 9110 - HTTP Semantics	Internet Standard	Interprétation exacte des requêtes et réponses HTTP	Un statut HTTP ne devient pas un verdict métier

L'adoption d'un standard n'en transfère pas automatiquement les garanties. Chaque intégration conserve un contrat explicite et un plafond de garantie.