KAMANGA

GUIDE GRATUIT · 14 PAGES · LEXIQUE KAMANGA

Le Lexique AIDD : 78 termes pour développer avec l'IA

Du LLM au harness agentique : tout le vocabulaire du développement assisté par IA, expliqué en une ligne par terme et organisé en 8 familles. Pour parler la même langue que vos outils, vos agents et vos pairs.

Kamanga Muludiki · 25 ans : développeur, tech lead, engineering leader, coach craft

kamanga.fr · Édition 2026

Le Lexique AIDD

78 termes pour développer avec l'IA

SOMMAIRE

Huit familles, une page chacune.

INTRODUCTION · 1 TERME

L'AIDD, et pourquoi son vocabulaire compte.

Le développement assisté par IA a produit un vocabulaire entier en quelques années. Le maîtriser n'est pas cosmétique : chacun de ces mots porte une pratique précise, et une équipe qui les partage évite les malentendus qui coûtent des sprints. Ce lexique définit chaque terme en une ligne, sans jargon creux.

> **LE TERME N°1 · AIDD**
>
> **AI-Driven Development** : développer AVEC l'IA comme acteur du cycle, pas comme autocomplete. L'IA cadre, implémente, teste et vérifie ; l'humain décide, arbitre et garde le merge.

Famille 01
Le modèle
Ce qu'est un LLM, ce qu'il voit, ce qu'il coûte. 9 termes.

Famille 02
Parler au modèle
Prompts, outils, formats de sortie. 9 termes.

Famille 03
L'agent et son harness
La boucle agentique et son environnement. 13 termes.

Famille 04
Orchestrer l'exécution
Multi-agents, isolation, résilience. 11 termes.

Famille 05
Vérifier et décider
La vérification adversariale et les gates humains. 8 termes.

Famille 06
La spec comme contrat
De la spec au coverage gate. 8 termes.

Famille 07
Mémoire et contexte
Ce qui survit à la session. 8 termes.

Famille 08
Mesurer et sécuriser
Evals et surfaces d'attaque. 11 termes.

77 termes dans les 8 familles, plus celui-ci : AIDD. Total : **78**. Lisez dans l'ordre, ou piochez la famille dont votre équipe a besoin cette semaine.

FAMILLE 01 · 9 TERMES

Le modèle : ce qu'il est, ce qu'il voit, ce qu'il coûte.

Tout part du modèle de langage. Ces neuf termes décrivent sa nature, ses limites et son économie : c'est le socle sur lequel tout le reste du lexique est construit.

TERME	DÉFINITION
LLM	Grand modèle de langage : un transformer entraîné à prédire le token suivant.
Token	Unité de lecture et de facturation du modèle (environ 3/4 de mot).
Context window	Quantité maximale de tokens visibles par le modèle en un appel.
Context rot	Dégradation de la précision du modèle à mesure que la fenêtre se remplit.
Température	Paramètre d'aléa de la génération : basse pour le code, plus haute pour l'idéation.
Cutoff	Date au-delà de laquelle le modèle n'a pas de connaissance d'entraînement.
Hallucination	Sortie plausible mais fausse ; se combat par citation, fact-check et vérification.
Embedding	Vecteur représentant le sens d'un texte, support de la recherche sémantique.
Fine-tuning	Ajustement des poids d'un modèle sur des données spécifiques.

À RETENIR

Tout se paie et se mesure en tokens : le prix des appels, la taille de la fenêtre, les budgets. La discipline de tokens est une compétence d'ingénierie à part entière.

FAMILLE 02 · 9 TERMES

Parler au modèle : prompts, outils, formats.

Un modèle ne vaut que par ce qu'on lui donne : des instructions claires, des exemples, des outils et un format de sortie exploitable. Ces neuf termes couvrent l'interface entre votre code et le modèle.

TERME	DÉFINITION
System prompt	Instructions de cadrage prioritaires ; hiérarchie system, puis user, puis outils.
Few-shot	Guider le modèle par des exemples plutôt que des descriptions.
Prompts as code	Prompts versionnés, relus et testés comme des modules.
Sortie structurée	Réponse du modèle contrainte par un schéma JSON validé.
Tool use	Capacité du modèle à appeler des fonctions définies par le développeur.
Streaming	Réception de la réponse token par token.
Prompt caching	Mise en cache des préfixes stables d'un prompt côté API (environ 10 fois moins cher).
RAG	Récupérer les passages pertinents d'un corpus et les injecter dans le prompt.
Chunking	Découpage d'un corpus en morceaux indexables (RAG).

> **LE RÉFLEXE PRO**
>
> La description d'un outil est un prompt, et un prompt est du code : il se versionne, se relit en pull request et se teste. Un prompt qui vit dans un ticket ou un chat est déjà perdu.

FAMILLE 03 · 13 TERMES

L'agent et son harness.

ÉTAPE 1
Percevoir
Lire le contexte et les résultats.

›

ÉTAPE 2
Raisonner
Choisir la prochaine action.

›

ÉTAPE 3
Agir
Appeler un outil.

›

ÉTAPE 4
Observer
Intégrer le résultat, boucler.

La boucle ci-dessus transforme un chatbot en agent. Le harness est tout ce qui l'entoure : la qualité d'un agent dépend moins du modèle que de l'environnement qu'on lui construit.

TERME	DÉFINITION
Agent	LLM + outils + objectif, exécutant une boucle perception, raisonnement, action.
Boucle agentique	Le cycle perceive, reason, act, observe répété jusqu'à l'objectif (pattern ReAct).
Subagent	Agent enfant avec contexte propre, outils restreints et modèle épinglé.
Orchestrateur	Agent qui coordonne et décide sans produire lui-même (« conductor, not a player »).
Harness	L'environnement d'exécution autour du modèle : outils, permissions, hooks, contexte.
Harness design	La discipline de conception de cet environnement : outils, contexte, garde-fous, feedback, mémoire.
Hook	Script déclenché sur le cycle de vie du harness (PreToolUse, SessionStart, Stop...).
Skill	Mode opératoire packagé et invocable du harness.
Rule	Standard de code chargé automatiquement sur les fichiers concernés (rules = comment écrire, skills = quoi faire).
MCP	Model Context Protocol : standard pour brancher des serveurs d'outils au harness.
Stack déclarative	Config projet lue par les agents (commandes, formatter, chemins) au lieu de supposer un layout.
Dual-harness	Deux harness en miroir sur le même repo (ex. Claude Code + Codex).
Fable-native	Conçu pour exploiter à fond le tier de modèle le plus capable.

FAMILLE 04 · 11 TERMES

Orchestrer l'exécution : plusieurs agents, zéro chaos.

Dès qu'une tâche dépasse un contexte, on la découpe entre plusieurs agents. Ces onze termes décrivent comment paralléliser sans collision, borner les coûts et survivre aux échecs.

TERME	DÉFINITION
Fan-out	Lancement de N agents en parallèle sur des sous-problèmes indépendants.
Workflow déterministe	Le contrôle de flux (boucles, votes, dédup) est du code ; le jugement seul revient au modèle.
Tier (S, F-lite, F-full)	Classe d'intensité d'une tâche : module le fan-out d'agents, jamais les passes exécutées.
Worktree	Copie de travail git isolée par tâche, prérequis du parallélisme agentique.
Model routing	Choisir le tier de modèle (Haiku, Sonnet, Opus, Fable) par tâche, pas par habitude.
Allow-list de modèles	Énumération des modèles autorisés en production, contre les configs dangereuses.
Idempotence	Relancer une opération ne duplique ni ne casse rien ; obligatoire pour tout ce qu'un agent peut re-tenter.
Fallback déterministe	Chemin non-LLM qui garantit le service quand l'IA échoue.
Circuit breaker	Après un échec de résolution d'un agent, basculer sur un générique et ne jamais re-tenter ce nom.
Drift guard	Test de parité byte-for-byte entre une copie nécessaire et sa source canonique.
Loop-until-dry	Itérer la découverte jusqu'à K rondes consécutives sans rien de nouveau.

> **LA RÈGLE D'OR**
>
> Le code décide, le modèle juge. Tout ce qui a une structure (partition, vote, dédup, diagramme) se code ; le modèle n'intervient que là où il faut du jugement.

FAMILLE 05 · 8 TERMES

Vérifier sans se fier : l'adversarial et les gates.

Un LLM est probabiliste : la qualité ne vient pas de la confiance mais de la vérification. Ces huit termes décrivent comment des agents se contredisent entre eux, et où l'humain garde la main.

TERME	DÉFINITION
Gate	Point de décision humaine dans un pipeline agentique.
DP (Decision Presentation)	Présenter une décision : contexte, options numérotées, recommandation, puis exécution immédiate.
Challenger	Agent adversarial qui cherche pourquoi NE PAS construire.
Skeptic	Agent adversarial qui tente de réfuter un finding ; seul ce qui survit est confirmé.
LLM-as-judge	Utiliser un modèle pour noter la sortie d'un autre.
Fact-forcing (GateGuard)	Refuser une action destructive tant que l'agent n'a pas constaté les faits (importeurs, rollback).
Conventional Comments	Format normalisé des findings de review (label, sévérité, rationale).
Success condition	Condition de fin de boucle EXÉCUTABLE (une commande), jamais déclarative.

> *« Dépenser du compute en vérification plutôt que multiplier les validations humaines : l'humain n'approuve plus chaque étape, il arbitre le goût aux moments stratégiques. »*

FAMILLE 06 · 8 TERMES

La spec comme contrat : écrire avant de générer.

Le SDD renverse le réflexe « on verra bien » : on écrit d'abord le contrat, l'IA implémente contre lui, et la vérification le prouve. Ces huit termes structurent la chaîne, de la spec à la preuve.

TERME	DÉFINITION
SDD	Spec-Driven Development : la spec est le contrat, l'IA implémente contre elle, la vérification la prouve.
Vibe coding	Développement à l'intuition sans spec ni vérification ; ce que l'AIDD discipliné remplace.
AC binaire	Critère d'acceptation vrai ou faux, jamais « à peu près » ; le contrat du cycle entier.
Coverage gate	Gate qui vérifie que chaque exigence est prouvée par un test avant de shipper.
Matrice des effets observables	Chaque effet déclaré dans la spec doit apparaître dans le diff, sinon finding bloquant.
Shadow areas	Scan adversarial des lacunes d'un artefact contre une taxonomie fermée.
ADR	Décision d'architecture datée et versionnée, lue par les agents autant que par les humains.
Décomposition axiale	Choix de l'axe primaire de variation d'un système ; se tromper d'axe duplique tout (piège N×M).

LE PIÈGE DE LA FAMILLE

Le vibe coding à l'échelle des agents

Sans spec, un agent produit vite... n'importe quoi, plus vite. Le volume amplifie l'erreur de cadrage.

ANTIDOTE Des critères d'acceptation binaires écrits AVANT la génération : chaque critère devient un test, chaque test arme le coverage gate.

FAMILLE 07 · 8 TERMES

Mémoire et contexte : ce qui survit à la session.

L'IA n'a pas de mémoire native entre deux sessions : l'AIDD lui en construit une. Ces huit termes couvrent la gestion de la fenêtre à court terme et la connaissance durable du projet.

TERME	DÉFINITION
Context engineering	La discipline de décider quoi mettre dans la fenêtre du modèle, et quoi en tenir dehors.
Context budget	Inventaire et pilotage de ce qui consomme la fenêtre (les serveurs MCP en tête).
Compaction	Résumé automatique ou manuel de l'historique pour libérer la fenêtre de contexte.
Condense (mode terse)	Mode de sortie compressé qui préserve code et avertissements verbatim.
Memory bank	Fichiers de connaissance projet durables, relus ou injectés à chaque session.
Instinct	Micro-régularité apprise « quand X, faire Y », pondérée par une confiance qui évolue.
Artifacts as state	L'avancement vit dans des fichiers versionnés, jamais dans la conversation.
Handoff	Document de passation (bilan + comment reprendre) pour continuer sans re-dériver le contexte.

LES 3 HORIZONS DE MÉMOIRE

Court terme : la fenêtre de contexte (une session). **Moyen terme** : les artefacts fichiers (un cycle de travail). **Long terme** : la memory bank (la vie du projet). Ce qui doit survivre à la session doit être un fichier.

FAMILLE 08 · 11 TERMES

Mesurer et sécuriser : evals et surfaces d'attaque.

Un système IA se mesure comme du code (avec des evals) et s'attaque autrement que du code (par ses prompts et son harness). Ces onze termes ferment la boucle : prouver la qualité, défendre le système.

TERME	DÉFINITION
Eval	Test exécutable du comportement d'un système IA (dataset + grader + métrique).
EDD	Eval-Driven Development : définir les evals avant d'implémenter la feature IA.
Grader	Ce qui note une sortie d'eval : code, règle, LLM-as-judge ou humain.
pass@k / pass^k	Au moins un succès sur k / tous les k réussissent (métriques d'eval).
Dogfooding	Construire l'outil avec l'outil lui-même ; la validation par l'usage réel.
Injection de prompt	Des données d'entrée qui contiennent des instructions ; se contre par délimiteurs et moindre privilège.
Trust boundary	Point où du texte non fiable re-entre dans un prompt ou sous les yeux de l'humain qui décide.
Firewall connaissance/exécution	Les skills de savoir produisent des artefacts à lire, jamais des modifications de code.
Dry-run par défaut	Une opération bulk n'exécute rien sans opt-in explicite ; elle imprime ce qu'elle ferait.
Vault	Stockage hors repo des secrets et données par utilisateur ; le repo garde le savoir versionnable.
Readiness	Garanties structurelles d'un repo (secrets, CI, hooks, ADR) avant d'y laisser travailler des agents.

> **LA RÈGLE DE PRUDENCE**
>
> Jamais un gate bloquant tenu par un LLM sans eval préalable. On commence en advisory, on mesure, puis on durcit.

SYNTHÈSE

Les 10 termes à maîtriser en premier.

78 termes, c'est une carte, pas un programme. Si vous débutez, ces dix-là suffisent pour comprendre 90 % des conversations AIDD et faire vos premiers choix outillés. Cochez au fur et à mesure.

- ☐ **Token** : parce que tout se paie et se mesure avec.
- ☐ **Context window** : parce que c'est la vraie limite de vos sessions.
- ☐ **System prompt** : parce que c'est votre premier levier de qualité.
- ☐ **Tool use** : parce que c'est ce qui transforme un chatbot en agent.
- ☐ **Agent** : parce que c'est l'unité de travail de l'AIDD.
- ☐ **Harness** : parce que la valeur durable est là, pas dans le modèle.
- ☐ **SDD** : parce que la spec écrite avant la génération change tout.
- ☐ **Gate** : parce que votre rôle d'humain se joue à ces moments-là.
- ☐ **Memory bank** : parce que redécouvrir chaque semaine coûte cher.
- ☐ **Eval** : parce que sans mesure, chaque changement de prompt est un pari.

> *« Un vocabulaire partagé est le premier outillage d'une équipe augmentée par l'IA : on ne pilote bien que ce qu'on sait nommer. »*

Ensuite, revenez famille par famille : chacune correspond à un chantier concret de votre process de développement.

À propos de l'auteur

KAMANGA Software Craftsman

L'AUTEUR

Kamanga Muludiki.

Coach craft et engineering leader international. 25 ans d'expérience et tous les rôles : développeur, tech lead, engineering manager, directeur technique, coach. Intervenu chez BNP Paribas, Crédit Agricole, Canal+, Agirc-Arrco et d'autres grandes structures, sur des transformations où l'enjeu économique du logiciel est réel.

Un double regard, business et technique. Diplômé d'un Executive MBA (Epitech Executive, Paris), j'aborde le logiciel comme un actif stratégique qui se pilote et qui doit produire de la valeur. Et je développe encore aujourd'hui, par passion : je comprends la technique dans le détail, et je sais ce que coûte (et ce que rapporte) un logiciel extrêmement bien fait, pérenne, qui dure dans le temps.

> *« La technologie est un outil au service du métier : la maîtriser, c'est garantir à l'entreprise des revenus solides et réguliers, et à ses clients une satisfaction durable. »*

Comment je peux vous aider

- **Conseil et accompagnement** : audit du process de développement, transformation des pratiques, coaching des équipes et des managers techniques.
- **Réalisation sur mesure** : conception et construction d'infrastructures techniques d'ingénierie augmentée par l'IA, marketplaces, plugins et outillage personnalisés.

Rester en contact

- LinkedIn : **@kamangacode** · linkedin.com/in/kamangacode
- Site : **kamanga.fr**
- Email : **herve@kamanga.fr**

LICENCE DE PARTAGE

© 2026 Hervemedia. Ce document peut être partagé à votre équipe dans sa totalité. Ne pas modifier sans accord.

Vous avez les mots.
Passez aux actes.

Ce lexique vous donne la carte. La suite se joue sur votre terrain : votre équipe, votre process, votre codebase. Quatre points de départ : deux pour les développeurs, deux pour les CTOs et leaders engineering.

DÉVELOPPEURS · DEUX FAÇONS DE PASSER UN CAP

EMA-Dev : auto-évaluation du développeur →

Situez votre profil face au rôle d'administrateur IA, et identifiez vos prochains gestes de progression. Gratuit.

Situer mon profil de développeur

Mentoring 1:1 →

Un blocage, un plafond de verre technique ? Un accompagnement individuel et un plan concret pour le franchir, avec un senior à vos côtés.

Coder comme un senior

CTOS ET LEADERS ENGINEERING · DEUX FAÇONS D'AVANCER

Session découverte (30 min) →

Échange gratuit sans engagement. On regarde la situation de votre équipe, on identifie le levier prioritaire.

Réserver mes 30 minutes

AI-Ready Engineering Checklist →

25 questions, 5 dimensions : score de maturité sur 25 et plan d'action 30/60/90 jours. Gratuit, 7 minutes.

Obtenir mon score AI-Ready

> *« La technologie est un outil au service du métier : la maîtriser, c'est garantir à l'entreprise des revenus solides et réguliers, et à ses clients une satisfaction durable. »*

POUR RESTER DANS LA BOUCLE

kamanga.fr LinkedIn @kamangacode herve@kamanga.fr