

LEÇON 14

Les patterns d'architecture du XXIe siècle

Architecture d'Entreprise

Les patterns d'architecture du XXIe siècle

Architecture d'Entreprise

V — LE LEADERSHIP

« L'architecture d'entreprise du XXIe siècle ne décrit pas un état stable du système d'information. Elle définit un potentiel de situation — la capacité de l'organisation à pivoter vite, à composer de nouvelles solutions à partir de briques autonomes, à intégrer l'intelligence artificielle sans la subir. » — Note d'un architecte senior, retour d'expérience sur un programme de transformation

« Le coût d'une donnée manquante n'est jamais le coût de la donnée. C'est le coût de la décision que vous n'avez pas pu prendre. » — Loi opérationnelle du data-driven, source anonyme, carnets de terrain

Vidéo d'introduction

[Vidéo à produire — Denise Neural, ~90 secondes, panorama des 4 patterns]

Le ticket de 72 heures

Septembre 2024, plateforme de services de Mutuelle Plus. Un adhérent appelle pour signaler une erreur sur son dernier remboursement. Le conseiller ouvre le dossier. Le dossier s'ouvre sur Gest+, le progiciel métier. Le progiciel cherche dans la base. La base prend huit secondes pour répondre — c'est acceptable. Le conseiller vérifie le remboursement, détecte une anomalie, veut la corriger. La correction nécessite de valider une exception, ce qui exige un workflow dans le module « Workflow » de Gest+. Le module est ralenti par les traitements batch du matin qui n'ont pas encore libéré la base. Le conseiller prend son déjeuner, revient, réessaie, échoue à nouveau. Trois jours plus tard, l'anomalie est corrigée. L'adhérent a déjà résilié.

Ce scénario n'est pas une exception chez Mutuelle Plus. C'est la norme : **72 heures** entre la déclaration d'un problème et sa résolution. Le marché — Alan, Alan, Malakoff Humanis, les nouveaux entrants digitaux — répond en moins de deux heures. Pour une mutuelle de 800 000 personnes protégées, chaque point de pourcentage de résiliation représente huit mille adhérents perdus. À 300 € de cotisation moyenne, c'est 2,4 millions d'euros de chiffre d'affaires volatilisé par an.

L'architecture d'entreprise des années 2010 — un progiciel central, des batchs nocturnes, des interfaces synchrones — n'est plus capable de soutenir la promesse client. Le XXIe siècle impose un changement de paradigme. Ce ne sont pas de nouveaux outils : ce sont de **nouvelles manières de penser le SI**. Quatre patterns redéfinissent ce qu'un architecte doit maîtriser aujourd'hui. Ils ne se substituent pas à TOGAF, BPMN, ArchiMate : ils en sont la **couche moderne**, celle qui permet à la méthode de produire de la valeur métier au rythme du marché.

Cette leçon présente ces quatre patterns — Event-Driven Architecture, Data Mesh, GenAI et FinOps — et montre comment l'architecte les intègre dans un cycle ADM contemporain, sans dogmatisme, avec un objectif clair : transformer la dette en potentiel.

1. Event-Driven Architecture — quand le SI devient un système nerveux

L'architecture événementielle part d'un constat simple : un SI traditionnel traite les demandes **à la demande** (*request-response*). Chaque interaction crée un couplage temporel entre l'émetteur et le récepteur : l'émetteur attend que le récepteur ait fini pour continuer. Quand le récepteur est ralenti, l'émetteur est bloqué. Quand le récepteur tombe, l'émetteur échoue.

L'architecture événementielle inverse la logique. Chaque changement d'état significatif dans le SI devient un **événement**, émis sur un canal, consommé par les acteurs intéressés — sans attendre, sans coupler. C'est la différence entre un appel téléphonique (synchrone, bloquant) et un email (asynchrone, non bloquant) — appliquée à l'échelle du SI.

Cinq composants fondamentaux.

Composant	Rôle	Exemple Mutuelle Plus
Producer	Détecte un changement d'état et émet l'événement	Le module « Sinistre déclaré » de Gest+, le CRM agent, le portail adhérent
Event	Message immuable notifiant un fait accompli	« Sinistre déclaré par l'adhérent X le 12/09 à 14h32, montant estimé 250 € »
Channel	Conduit logique transportant l'événement	Un topic Kafka, une file Azure Service Bus, un flux Kinesis
Processor	Composant consommateur exécutant la logique métier	Module de calcul de remboursement, module de détection de fraude, module CRM
Broker	Pivot gérant la distribution, la persistance, le routage	Kafka, RabbitMQ, AWS EventBridge

Cette décomposition n'est pas qu'organisationnelle : elle est **politique**. Elle permet à l'AE d'attribuer à chaque direction métier la propriété d'un sous-ensemble d'événements, sans exiger qu'elle comprenne le SI global. Le DAF n'a pas besoin de comprendre le calcul de remboursement ; il a besoin que l'événement « Sinistre déclaré » soit émis de manière fiable. C'est cette **asymétrie de connaissance** qui rend l'EDA compatible avec des organisations en silos.

Deux grandes familles se distinguent.

L'**Event Sourcing** persiste chaque changement d'état comme un événement, et reconstruit l'état courant par rejeu des événements. C'est l'approche privilégiée pour les SI où l'**audit** est critique : on peut reconstruire l'état du SI à n'importe quel point du passé. Chez une mutuelle, c'est précieux : en cas de contrôle ACPR, on peut répondre à « quel était l'état du dossier de cet adhérent le 12 mars 2023 à 14h ? » sans dépendre d'une hypothétique sauvegarde.

L'**Event Streaming** traite des flux continus d'événements pour réagir **en temps réel** à des motifs. On ne cherche pas à reconstruire le passé ; on cherche à détecter des patterns dans le présent. Chez Mutuelle Plus, c'est ce qui permet d'identifier une fraude en cours (« trois déclarations de sinistres inhabituelles sur les six dernières heures pour le même numéro de contrat ») ou de signaler un dépassement de plafond de garantie en temps réel.

Ces deux familles ne sont pas mutuellement exclusives. Une architecture EDA mature combine souvent les deux : Event Streaming pour la détection temps réel, Event Sourcing pour l'auditabilité.

L'arbitrage central : remplacer un **ESB traditionnel** (Enterprise Service Bus) par un **Message Broker moderne** (Kafka, Pulsar, NATS). L'ESB traditionnel centralise une logique métier complexe, fragile, propriétaire ; il devient le point de défaillance unique. Le Broker moderne privilégie la scalabilité horizontale : on ajoute des consommateurs sans modifier les producteurs, on gère la backpressure sans bloquer les émetteurs, on garantit la persistance sans coupler les flux. C'est cette dissociation qui permet à Mutuelle Plus d'ajouter une nouvelle campagne de santé en quelques semaines sans toucher au progiciel Gest+.

En termes ArchiMate 4, le flux se modélise ainsi : un **Business Event** (« Customer Order ») déclenche un **Business**

Process (« Contract Creation ») qui réalise un **Business Object** (« Active Policy »), émis via une **Flow Relationship** vers les consommateurs intéressés. Les interfaces entre le legacy (Gest+) et les nouveaux services utilisent des messages standardisés (JSON, Avro) pour assurer l'interopérabilité sans modification intrusive du cœur de système.

2. Data Mesh — quand la donnée devient un produit de domaine

L'architecture d'entreprise des années 2010 concevait la donnée comme un **actif central** à maîtriser par la DSI : un Data Warehouse, un MDM, une équipe de data engineers en position de gatekeeper. Ce modèle a deux limites. Premièrement, il crée un goulot d'étranglement : chaque nouvelle question métier nécessite un projet data, avec une équipe data, dans une file d'attente data. Deuxièmement, il déresponsabilise les métiers : les directions opérationnelles consomment la donnée, mais ne la produisent pas avec la qualité attendue. Le résultat, chez Mutuelle Plus comme ailleurs, est un patrimoine data centralisé mais dégradé : doublons dans Gest+, tableurs en agence, données de santé fragmentées entre quinze systèmes.

Le **Data Mesh**, théorisé par Zhamak Dehghani en 2019, propose un changement de paradigme : la donnée n'est plus un actif centralisé, c'est un **produit distribué** sous la responsabilité des domaines métier.

Quatre principes.

- 1. Domain Ownership.** Chaque domaine métier (Santé Individuelle, Prévoyance Collective, Gestion des Réclamations, etc.) possède ses données. La DSI ne « gère » plus la donnée ; elle fournit l'infrastructure qui permet aux domaines de la gérer eux-mêmes. *Conséquence* : les responsables métier sont responsables de la qualité à la source, pas les data engineers en bout de chaîne.
- 2. Data as a Product.** Chaque ensemble de données exposé par un domaine est un « produit » avec un propriétaire nommé, un contrat d'interface (schema, SLA, latence), une documentation utilisateur. *Conséquence* : un produit data est aussi adressable qu'une API ; il peut être consommé par d'autres domaines sans négociation bilatérale.
- 3. Self-Serve Platform.** L'infrastructure data (pipelines, catalogue, gouvernance, monitoring) est fournie comme une plateforme interne que les domaines utilisent sans expertise technique spécialisée. *Conséquence* : les domaines publient et consomment de la donnée sans dépendre d'une équipe data centrale.
- 4. Federated Governance.** Les standards d'interopérabilité (formats, identifiants, règles de sécurité, conformité RGPD) sont définis globalement et appliqués automatiquement. *Conséquence* : la cohérence globale est garantie par la plateforme, pas par un comité manuel.

Cette architecture produit un résultat stratégique majeur : **résolution de la dette d'intégration par les Data Contracts**. Quand un domaine modifie la structure de ses données, le contrat de données expose le changement, et les consommateurs peuvent s'adapter sans casser. C'est la résolution directe du problème que Gest+ connaît aujourd'hui : modifier un champ métier dans Gest+ nécessite de modifier quinze rapports en aval.

Matrice de décision : quand adopter le Data Mesh, quand conserver un MDM centralisé ?

Critère	MDM centralisé	Data Mesh
Maturité data de l'organisation	Moyenne, IT-centric	Haute, culture de domaine établie
Nombre de domaines métier	Faible (< 5)	Élevé (> 10)
Besoin d'agilité	Standard	Critique (transformation continue)
Compétences data disponibles	Centralisées	Distribuées dans les domaines
Culture d'ownership	Faible	Forte

Mutuelle Plus, avec ses quinze domaines opérationnels et son projet de refonte du SI Décisionnel, est aujourd'hui à la **frontière** : un MDM centralisé pourrait encore tenir trois à cinq ans, mais un Data Mesh serait préférable pour soutenir la transformation à cinq ans. L'AE recommande une trajectoire **hybride** : MDM centralisé sur le domaine « contrats » (haute criticité, faible maturité), Data Mesh sur les domaines périphériques (marketing, satisfaction, observatoire).

3. GenAI dans la cartographie applicative — l'architecte augmenté

L'intelligence artificielle générative — Large Language Models, Retrieval-Augmented Generation, agents autonomes — n'est pas un gadget pour l'architecte d'entreprise. C'est un **accélérateur de productivité documentaire** et un **levier de souveraineté sur le patrimoine d'architecture**.

Concrètement, un LLM couplé à un RAG sur le référentiel d'architecture permet :

- **L'analyse d'impact en langage naturel.** Poser une question du type « quelles sont les applications impactées par la migration du module Auth de Gest+ ? » et obtenir une réponse structurée avec les dépendances identifiées. Là où l'AE passait deux jours à feuilleter le référentiel, il obtient une réponse en deux minutes — qu'il valide ensuite par revue humaine.
- **La détection de redondances fonctionnelles.** Soumettre au LLM la liste des applications métiers pour identifier les recouvrements : « Ces deux applications font probablement la même chose, à vérifier avec les métiers. » Mutuelle Plus a ainsi identifié trois doublons fonctionnels entre Gest+ et des outils locaux développés par les services opérationnels.
- **La génération de fiches applicatives.** À partir de la documentation existante (souvent parcellaire, désordonnée), le LLM produit une fiche normalisée : description, fonctionnalités, dépendances, propriétaire, contraintes. L'AE économise huit à quinze jours-homme par cycle d'enrichissement du référentiel.
- **L'assistance à la modélisation ArchiMate.** Demander au LLM de produire un draft de diagramme ArchiMate à partir d'un brief métier en langage naturel. Le résultat est à 70 % exploitable ; les 30 % restants sont à reprendre pour la cohérence stylistique.

Ces gains sont réels. Mais ils viennent avec **trois risques structurels** que l'AE doit encadrer.

Risque 1 — Confidentialité. Un LLM mal configuré peut exposer des données de santé (catégorie spéciale RGPD) dans ses réponses, ou les transmettre à des serveurs tiers. *Atténuation* : modèles privés isolés (Azure OpenAI en mode privé, Mistral self-hosted), hébergement des données du référentiel dans un périmètre cloisonné, audit des flux par le RSSI.

Risque 2 — Hallucinations. Un LLM peut produire des schémas techniques erronés, inventer des dépendances qui n'existent pas, ou proposer des solutions non conformes aux standards internes. *Atténuation* : validation humaine obligatoire par un architecte senior, journalisation des suggestions, mesure de la précision sur un échantillon de validation avant déploiement.

Risque 3 — Dette technique IA. Le code généré, les configurations générées, les schémas générés peuvent être non maintenables, inintelligibles, ou non conformes aux standards de développement. *Atténuation* : standards de codage stricts appliqués au code généré, revue systématique par un Lead Dev, gouvernance par l'Architecture Review Board.

Cadres de gouvernance à appliquer. Le **NIST AI Risk Management Framework** propose quatre fonctions (Govern, Map, Measure, Manage) pour évaluer et mitiger les risques. L'**EU AI Act** (entrée en application progressive depuis 2024) classe les usages par niveau de risque et impose des obligations proportionnelles : un usage en « risque limité » (assistant documentaire) requiert transparence ; un usage en « risque élevé » (décision automatique sur dossier adhérent) requiert certification, auditabilité, supervision humaine.

Pour Mutuelle Plus, l'usage de GenAI en documentation d'architecture relève du **risque limité** sous réserve d'encadrement : données hébergées en privé, validation humaine systématique, audit annuel. Un usage futur pour la détection de fraude ou l'aide à la décision sur les dossiers rentrerait dans le **risque élevé**, avec des obligations sensiblement plus lourdes.

4. FinOps — quand l’efficacité financière devient une exigence d’architecture

Le FinOps — contraction de « Finance » et « DevOps » — est l’ensemble des pratiques qui permettent d’**optimiser en continu les coûts Cloud** en impliquant équipes techniques, financières et métiers dans une gouvernance partagée. Ce n’est pas de la réduction budgétaire : c’est de l’**allocation intelligente** des ressources pour maximiser la valeur par euro investi.

Trois phases structurent la maturité FinOps d’une organisation.

Phase 1 — Inform. Rendre visibles les coûts Cloud par domaine métier, par application, par équipe. Sans cette visibilité, aucune optimisation rationnelle n’est possible. Concrètement : dashboards de cost allocation, tags par domaine, intégration des données de facturation dans le référentiel d’architecture.

Phase 2 — Optimize. Identifier les ressources sous-utilisées, négocier les engagements (Reserved Instances, Savings Plans), ajuster les architectures pour réduire le gaspillage. L’AE intervient ici en challengeant les choix de sizing dans les revues d’architecture : « *Cette instance est dimensionnée pour 10 000 utilisateurs ; tu en as 800. Pourquoi ?* »

Phase 3 — Operate. Intégrer la dimension financière dans la gouvernance continue : budgets par équipe, alertes de dépassement, arbitrages réguliers entre valeur métier et coût d’infrastructure. C’est l’état mature où l’efficacité financière est un réflexe d’équipe.

Trois indicateurs clés pour l’architecte.

KPI	Description	Impact stratégique
Coût par transaction	Coût Cloud par opération métier (remboursement, adhésion, réclamation)	Si le coût IT dépasse la marge des contrats, le modèle économique s’effondre
Idle ratio	Part des ressources provisionnées mais non consommées	Réduction directe du gaspillage des fonds propres
Commitment discount	Remises obtenues par engagement de durée	Optimisation de la capacité d’investissement

Intégration dans l’Architecture Review Board. Chaque design cible présenté à l’ARB doit désormais inclure une checklist financière :

- **Unit Economics** : le coût marginal IT par nouvel adhérent est-il inférieur au coût d’acquisition client ?
- **Cloud Exit Strategy** : existe-t-il un plan de réversibilité documenté pour éviter le verrouillage tarifaire (vendor lock-in) ?
- **Élasticité** : le design permet-il une extinction automatique des ressources hors heures ouvrées ?
- **Coût de la dette** : quelle est la part du coût courant imputable à la dette technique ?

Ces questions ne sont pas optionnelles. Elles transforment l’ARB d’un comité technique en un **comité d’arbitrage stratégique**, où les décisions intègrent simultanément la cohérence architecturale, la conformité, et l’efficacité économique.

5. L’entreprise fractale — la vision cible à cinq ans

Les quatre patterns précédents ne sont pas indépendants. Ils se renforcent mutuellement dans une vision cible que l’on peut qualifier d’**entreprise fractale** : une organisation où l’agilité globale se reproduit à toutes les échelles — du

portefeuille d'affaires jusqu'au composant technique.

Dans cette vision à cinq ans, Mutuelle Plus n'est plus un progiciel central entouré de satellites. C'est un **écosystème de produits** :

- **Couche réactive (EDA)** : tous les événements métier significatifs sont émis en temps réel. Un sinistre déclaré déclenche en parallèle le calcul de remboursement, la détection de fraude, la mise à jour du CRM, la notification à l'adhérent, l'enregistrement comptable. Les traitements batch disparaissent.
- **Couche data distribuée (Data Mesh)** : chaque domaine (Santé Individuelle, Prévoyance, Réclamations, Marketing) expose ses données comme un produit, avec un contrat d'interface et une SLA. Les doublons disparaissent. Les délais d'accès à la donnée chutent de six mois à trois semaines.
- **Couche cognitive (GenAI)** : l'architecte dialogue avec le référentiel en langage naturel. Les nouveaux produits sont qualifiés en deux semaines au lieu de six mois. La documentation est vivante.
- **Couche efficiente (FinOps)** : chaque euro Cloud est tracé, alloué, optimisé. La marge de solvabilité est restaurée. Les réinvestissements dans la transformation sont autofinancés par les gains de productivité.

Cette vision n'est pas un slogan. C'est ce qui permet à une mutuelle de 800 000 adhérents de **lancer une nouvelle garantie santé en six semaines** au lieu de douze mois, avec une marge de solvabilité sécurisée, une conformité RGPD garantie, et une expérience adhérent en temps réel.

L'architecte d'entreprise est le pilote de cette transformation. Il ne dessine pas le SI cible ; il **établit les conditions** pour que l'organisation se transforme elle-même, à son rythme, avec ses contraintes, en s'appuyant sur les quatre patterns comme briques modulaires.

6. Cinq principes opérationnels pour l'architecte du XXI^e siècle

De ces quatre patterns se dégagent cinq principes. Ils ne remplacent pas ceux des leçons précédentes ; ils les complètent pour le contexte contemporain.

Premier principe — Le potentiel de situation prime sur l'état stable. *Ne pas viser une cible figée, viser une capacité d'évolution.* L'AE ne cherche plus à figer l'architecture cible ; il cherche à maximiser la **capacité de l'organisation à pivoter**. L'EDA et le Data Mesh sont des patterns qui produisent du potentiel. Le MDM centralisé et l'ESB synchrones produisent de la rigidité. La question n'est plus « quelle est la cible ? » mais « quelle est la vitesse de réorientation ? »

Deuxième principe — Découpler temporellement, coupler sémantiquement. *Le bon couplage n'est pas l'absence de couplage, c'est le couplage intelligent.* L'EDA découple temporellement (Producteur et Consommateur ne s'attendent pas) mais couple sémantiquement (ils partagent le format de l'événement, le contrat d'interface). Le Data Mesh découple organisationnellement (chaque domaine est autonome) mais couple contractuellement (Data Contracts standardisés). L'AE conçoit les contrats, pas les composants.

Troisième principe — Distribuer la responsabilité, centraliser la gouvernance. *Le Data Mesh n'est pas l'anarchie data.* Chaque domaine est propriétaire de ses produits data, mais la plateforme fédérale garantit les standards d'interopérabilité, de sécurité, de conformité. La gouvernance est globale ; l'exécution est locale. C'est l'application du principe de subsidiarité à l'architecture.

Quatrième principe — Instrumenter avant d'optimiser. *On ne optimise pas ce qu'on ne mesure pas.* Avant d'optimiser les coûts Cloud (FinOps), il faut les mesurer (Phase 1 Inform). Avant d'optimiser les workflows EDA, il faut observer les patterns d'émission et de consommation. Avant de détecter les redondances par GenAI, il faut disposer d'un référentiel d'architecture propre. L'AE qui saute à l'optimisation sans instrumentation crée de la complexité sans valeur.

Cinquième principe — L'IA augmente l'architecte, ne le remplace pas. *Le LLM est un copilote, pas un pilote.* L'AE qui délègue ses jugements à un modèle prend des décisions qu'il ne peut plus justifier. L'AE qui utilise le modèle comme assistant de production — pour générer des drafts, détecter des patterns, suggérer des options — garde la maîtrise de la décision finale. La validation humaine n'est pas une perte de temps ; c'est le **garde-fou de la responsabilité**.

7. Application — Quel pattern pour quel problème ?

Trois exercices à conduire dans les 30 jours.

Exercice 1 — Diagnostic de votre dette d'architecture contemporaine. Pour chacun des quatre patterns, notez votre niveau de maturité actuel sur une échelle de 0 à 3 (0 = absent, 1 = embryonnaire, 2 = en cours, 3 = industrialisé).

- EDA : votre organisation émet-elle des événements métier en temps réel ? Lesquels ? Quel broker ?
- Data Mesh : vos domaines métier possèdent-ils leurs données ? Existe-t-il des Data Contracts formalisés ?
- GenAI : utilisez-vous un LLM pour augmenter votre productivité d'architecte ? Avec quelles garde-fous ?
- FinOps : connaissez-vous le coût par transaction métier de votre SI ? Savez-vous qui consomme quoi ?

Exercice 2 — Choisir un pattern prioritaire. Sélectionnez le pattern où votre dette est la plus critique. Décrivez en une page : le contexte (quel problème métier ?), la cible à 12 mois (qu'est-ce qui aurait changé ?), les trois premières étapes (par quoi commencer ?), le critère de succès mesurable.

Exercice 3 — Cartographier un flux EDA. Choisissez un processus métier réel de votre organisation. Modélisez-le sous forme événementielle : quels sont les producteurs, les événements significatifs, les canaux, les consommateurs, le broker. Comparez avec l'architecture actuelle (synchrone, batch, ESB). Chiffrez le gain de latence attendu.

Cette leçon n'enseigne pas à devenir ingénieur Kafka ou data scientist. Elle enseigne à l'architecte à dialoguer avec ces expertises, à en comprendre les apports et les limites, à arbitrer entre les patterns en fonction du contexte.

Encadré Au lecteur / À l'animateur

Question d'ouverture : « Dans votre organisation, quel est le délai entre un événement métier (commande, sinistre, adhésion) et sa prise en compte par tous les systèmes qui devraient le connaître ? Pourquoi est-il si long ? »

Profils à observer chez l'apprenant :

- **Le technophile qui veut tout réécrire en Kafka.** Confond modernisation et disruption. *Question : « Quel est le coût de migration de votre ESB vers un broker moderne, et combien de projets métier cela retarderait-il ? »*
- **Le conservateur qui refuse les patterns modernes.** Considère que TOGAF et BPMN suffisent. *Question : « Comment votre organisation lance-t-elle un nouveau produit aujourd'hui ? Combien de temps ? Pourquoi ? »*
- **Le data-centrique qui croit que la donnée résout tout.** Veut centraliser toute la donnée avant de la distribuer. *Question : « Combien de temps avez-vous passé sur votre dernier projet data centralisé ? Quel est le ROI ? »*
- **L'IA-enthousiaste qui délègue tout au LLM.** Considère que l'IA va remplacer l'architecte. *Question : « Quel jugement architectural pouvez-vous déléguer sans perdre votre responsabilité ? »*

Pièges d'animation :

- Ne pas laisser la discussion dériver vers les outils (Kafka vs RabbitMQ, OpenAI vs Mistral). Les patterns sont neutres vis-à-vis de l'implémentation.
- Ne pas idéaliser les patterns. Tous ont des coûts, des limites, des prérequis organisationnels. Un EDA mal gouvernée produit plus de chaos qu'une SOA bien gérée.
- Rappeler que ces patterns sont complémentaires, pas substitutifs. Le Data Mesh ne remplace pas le MDM dans tous les cas ; le FinOps ne remplace pas la discipline budgétaire.

Si profil d'apprenant ≠ novice : ouvrir la discussion sur les patterns non couverts ici (architectures cellulaires,

plateformes de données unifiées, edge computing, IA agentique). Identifier le pattern émergent que l'apprenant considère comme stratégique pour son contexte, et l'aider à l'analyser avec la grille de cette leçon.

Ce que le cursus vous demande maintenant

Vous avez maintenant traversé quatorze leçons — douze leçons techniques plus deux leçons sur le leadership et la modernité. Vous avez les savoirs (les frameworks), les savoir-faire (la méthode, les patterns), et les savoir-être (la posture politique). Le cursus s'arrête ici sur les fondamentaux. La suite est votre propre trajectoire.

Trois engagements pour les 60 prochains jours :

- **Cartographier** la maturité de votre organisation sur les quatre patterns de cette leçon.
- **Choisir** un pattern prioritaire et rédiger la première page d'un dossier de modernisation.
- **Initier** une conversation GenAI structurée avec votre DSI ou votre RSSI, sur un cas d'usage limité et encadré.

L'architecture d'entreprise du XXIe siècle n'est pas une discipline figée. C'est une **conversation continue** entre l'organisation et son écosystème technologique. L'architecte est celui qui tient la conversation, qui la structure, qui la fait avancer — sans jamais prétendre en connaître le dernier mot.

Quiz — 18 questions pour valider votre compréhension

Trois séries de six questions. Prenez un crayon, écrivez vos réponses dans l'espace prévu. Score indicatif : 14/18 minimum pour valider la leçon.

A — COMPREHENSION

A1 Quels sont les cinq composants fondamentaux d'une architecture événementielle (EDA) ? Quel rôle joue le broker ?

A2 Quelle est la différence entre Event Sourcing et Event Streaming ? Dans quels cas d'usage privilégier chacun ?

A3 Quels sont les quatre principes du Data Mesh ? Pourquoi distinguent-ils ownership et gouvernance ?

A4 Quels sont les trois risques majeurs de l'usage de GenAI dans la cartographie applicative ? Quelles atténuations proposer ?

A5 Quelles sont les trois phases de maturité FinOps ? Quel est le rôle de l'architecte dans chacune ?

A6 Qu'est-ce qu'une « entreprise fractale » ? En quoi cette vision cible diffère-t-elle d'une architecture cible traditionnelle ?

B — AUTO-EVALUATION

B1 Dans ton organisation, quels sont les événements métier émis en temps réel aujourd'hui ? Lesquels sont encore en batch nocturne ?

B2 Combien de domaines métier dans ton organisation possèdent réellement leurs données ? Combien sont en accès libre via un catalogue ?

B3 As-tu déjà testé un LLM sur ton référentiel d'architecture ? Quel gain de temps as-tu observé ? Quelles erreurs as-tu détectées ?

B4 Connais-tu le coût par transaction métier de ton SI ? Le coût par adhérent / client / usager ? Comment est-il mesuré ?

B5 Combien de fois as-tu été bloqué par un couplage temporel fort (Producteur qui attend Consommateur) dans les six derniers mois ?

B6 Combien de jours sépare, dans ton organisation, un événement métier (commande, sinistre, adhésion) de sa prise en compte par tous les systèmes ?

C — ENGAGEMENTS A 30/45/60 JOURS

C1 Cartographier ma maturité sur les quatre patterns (EDA, Data Mesh, GenAI, FinOps) sur une échelle de 0 à 3. (Échéance : J+14.)

C2 Choisir le pattern où ma dette est la plus critique et rédiger une première page de dossier de modernisation (contexte, cible 12 mois, 3 premières étapes, KPI). (Échéance : J+30.)

C3 Modéliser un processus métier réel sous forme événementielle (5 composants EDA) et chiffrer le gain de latence attendu. (Échéance : J+30.)

C4 Identifier trois domaines métier candidats au Data Mesh et trois domaines où un MDM centralisé reste justifié. (Échéance : J+30.)

C5 Initier une conversation structurée avec mon DSI / RSSI sur un cas d'usage GenAI limité et encadré (données, validation, audit). (Échéance : J+45.)

C6 Mettre en place un dashboard de cost allocation Cloud par domaine métier dans les 60 jours (FinOps Phase 1). (Échéance : J+60.)
