SCALEWAY CLOUD

IDEO-LAB - Guillaume Oneill

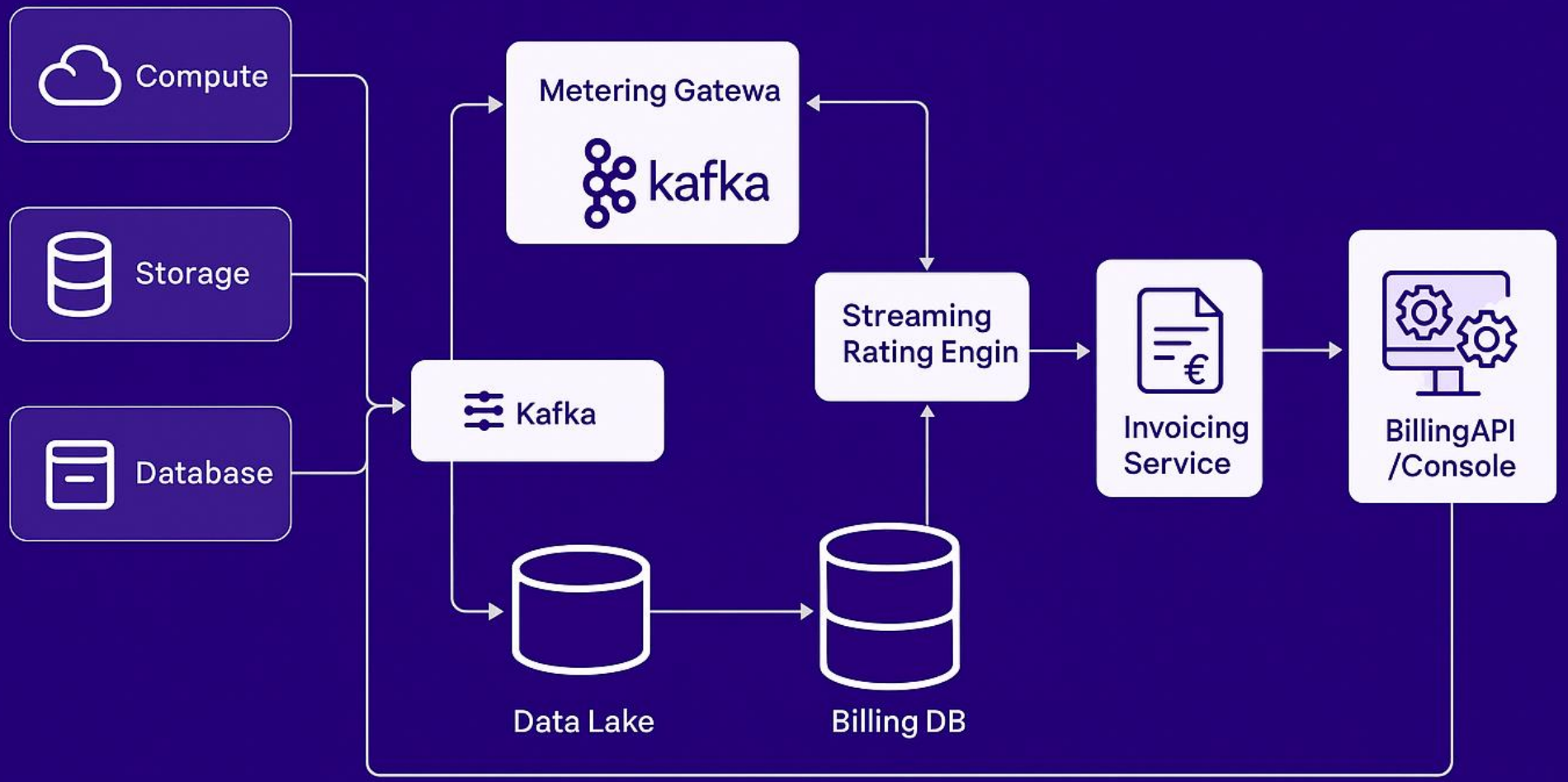
AVRIL 2025 – V 1.2

SCALEWAY EN EUROPE

AGENDA GENERAL

- PRESENTATION GLOBALE SCALEWAY
- OFFRE DE SERVICES
- CORE BILLING
- MISE EN OEUVRE
- CONCLUSION

SCALEWAY

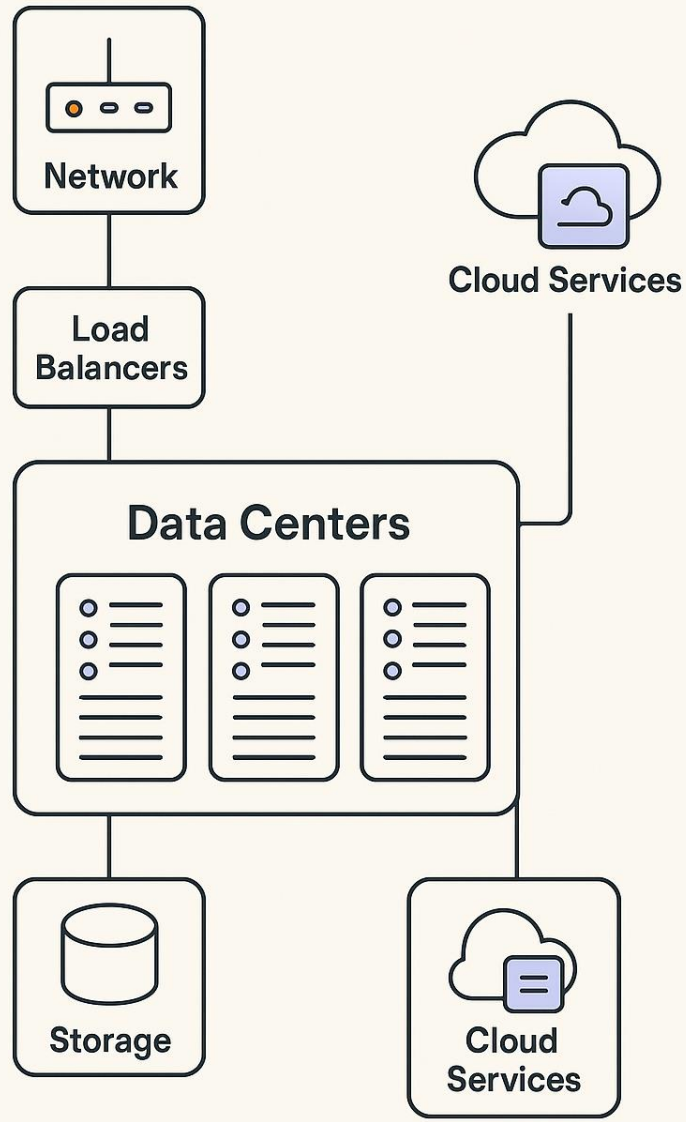


SCALEWAY ARCHITECTURE

IDEO·LAB

- Présentation de Scaleway
- Architecture et Data Centers
- Certification & Souveraineté
- Gamme de Service Cloud

Data Centers



Présentation de Scaleway

- **Scaleway** (anciennement Online SAS) est un fournisseur français d'infrastructure cloud, filiale d'Iliad, fondé en 1999 par Xavier Niel Wikipédia +13.
- Il propose des services de **compute (IaaS)**, de **serveurs dédiés**, d'**hébergement web**, d'**enregistrement de noms de domaine** (BookMyName) et de **colocation** via Scaleway Datacenter Wikipédia +1.
- Scaleway se positionne comme un **acteur du cloud souverain européen**, rival direct d'OVHcloud, avec environ 100 M€ de chiffre d'affaires en 2022, malgré des pertes dues aux investissements cloud et IA Wikipédia +8.

Architecture et datacenters

- Les services cloud sont hébergés dans des **datacenters en Île-de-France** exploités par **OpCore** (séparé depuis 2023) Wikipédia.
- Principaux centres :
  - **DC2** (Vitry-sur-Seine) : rénové entre 2009-2012.
  - **DC3** (Vitry-sur-Seine) : construit en 2012, atteint la saturation en 2016.
  - **DC4** (Paris 15e) : ancien abri antiatomique, héberge l'offre **C14 Cold Storage**.
  - **DC5** (Saint-Ouen-l'Aumône) : refroidissement adiabatique, PUE proche de 1, innovation brevetée Wikipédia +2.
- Scaleway dispose également de datacenters à **Amsterdam (AMS1)** et **Varsovie (WAW1)** pour une présence européenne Wikipédia +13.

Certifications & souveraineté

- Conformité **RGPD**, certification **ISO/IEC 27001**, et **HDS** pour les données de santé Scaleway +2.
- Engagement dans le cloud souverain :
  - Alliance avec **Atempo** pour une solution sécurisée et hébergée en France Wikipédia +4.
  - Membre fondateur du projet **Gaia-X**, mais retrait en 2021 Wikipédia.
  - En démarche pour la qualification **SecNumCloud** de l'ANSSI (2024) Wikipédia +5.

Gamme de services cloud

1. Compute & Bare-Metal

- **Instances virtuelles** : VMs x86 ou ARM, adaptées aux besoins de dev/test jusqu'à la production xraise.ai +6.
- **Elastic Metal** : bare-metal intégrable à l'écosystème public cloud – hybride possible Scaleway +7.
- **Mac mini M1 (Apple Silicon as a Service)** : idéal CI/CD iOS/macOS, rare en Europe Wikipédia +1.
- Des configurations bare-metal avec Xeon ou EPYC, déployables rapidement, à partir de **≈ 8,99 €/mois** TechRadar.

GAMME DE SERVICES

GAMME DE SERVICES

- Compute & Bare Metal
- Stockage
- Bases de données Managées
- Kubernetes
- ServerLess
- Réseaux et Sécurité
- AI & GPU
- Web Hosting
- Documentation & API

2. Stockage

- **Object Storage**, compatible S3, avec 75 GB d'egress gratuit, stockage froid (type Glacier) xraise.ai +2.
- **Block Storage** à faible latence, volumes performants Scaleway +1.
- **C14 Cold Storage** (DC4) pour archivage longue durée Wikipédia.

3. Bases de données managées

- Postgres, MySQL, Redis, MongoDB®, avec sauvegardes, scalabilité et haute disponibilité xraise.ai.

4. Kubernetes

- **Kapsule** (mono-cloud) et **Kosmos** (multi-cloud) : clusters Kubernetes managés, plans gratuits ou payants xraise.ai +5.

5. Serverless

- **Functions as a Service**, **Containers serverless** (scale-to-zero) avec registre intégré ou externe A Java geek.
- Idéal pour des déploiements légers ou événementiels.

6. Réseaux & Sécurité

- **IAM** avec contrôle d'accès RBAC pour collaborateur interne/externe Scaleway.
- Réseau autonome (AS12876), interconnexions France-IX, Equinix-IX, AMS-IX Wikipédia +2.

7. AI & GPU

- Accès aux GPU NVIDIA (H100, L40S, L4) pour workloads ML/IA xraise.ai.
- **Managed Inference** : déploiement de modèles, autoscaling, API compatibles OpenAI Scaleway +1.

8. Web hosting

- Plans **cPanel hébergés** : Lite, Personal, Professional, Ecommerce avec vCPU, RAM, SSD, e-mails, bases illimitées – de 5,99 à 29,99 € HT/mois Scaleway.

9. Documentation & API

- Documentation exhaustive : guides, API/CLI, tutoriels, exemples pour Secret Manager, AI, Bare Metal, etc. Scaleway.

Atouts & limites (selon avis tierce partie)

Avantages :

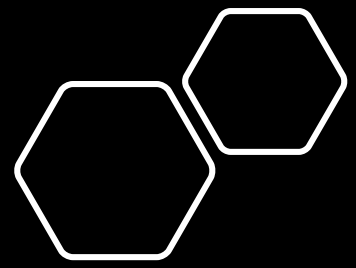
- Tarification transparente, compétitive.
- Respect de la souveraineté EU.
- Infrastructure durable et écoresponsable.
- Offre unique Mac mini et GPU haut de gamme.
- Bons crédits pour startups (jusqu'à 36 000 €) xraise.ai Wikipédia.

Inconvénients :

- Moins de zones géographiques couvertes.
- Écosystème moins mature que AWS/GCP.
- Support variable selon les offres.
- Redimensionnement de VM parfois non trivial via UI xraise.ai +1.
- Pas de compliance HIPAA (USA) xraise.ai.

Synthèse technique

Domaine	Points clés
Infrastructure	Datacenters en France, Pays-Bas, Pologne ; exploitation OpCore
Compute	VMs (x86, ARM), bare-metal, Mac mini M1
Stockage	Objet S3, block, stockage froid C14
DB Managées	Postgres, MySQL, Redis, MongoDB®
Kubernetes & Serverless	Kapsule/Kosmos, fonctions, containers serverless
AI / GPU	NVIDIA H100, L40S, L4 + inference managée
Réseau & Sécurité	IAM, RGPD, ISO 27001, HDS, SecNumCloud en projet
Web Hosting	Plans cPanel ciblant PME et sites professionnels
Documentation	API/CLI, guides pour presque tous les services
Différenciateurs	Souveraineté EU, durabilité, transparence, spécialisation niche



OFFRES DE SERVICES

OFFRES DE SERVICES

1. **Development Instances** (DEV1 & GP1) — versions "entrée de gamme" / usage développement
2. **General Purpose Instances** — usage de production classique
3. **Compute Optimized Instances** — calcul intensif, haute performance
4. **Memory Optimized Instances** — mémoire importante pour big data
5. **Specialized Instances (Workload-Optimized)** — selon les besoins spécifiques

Comparatif Synthétique

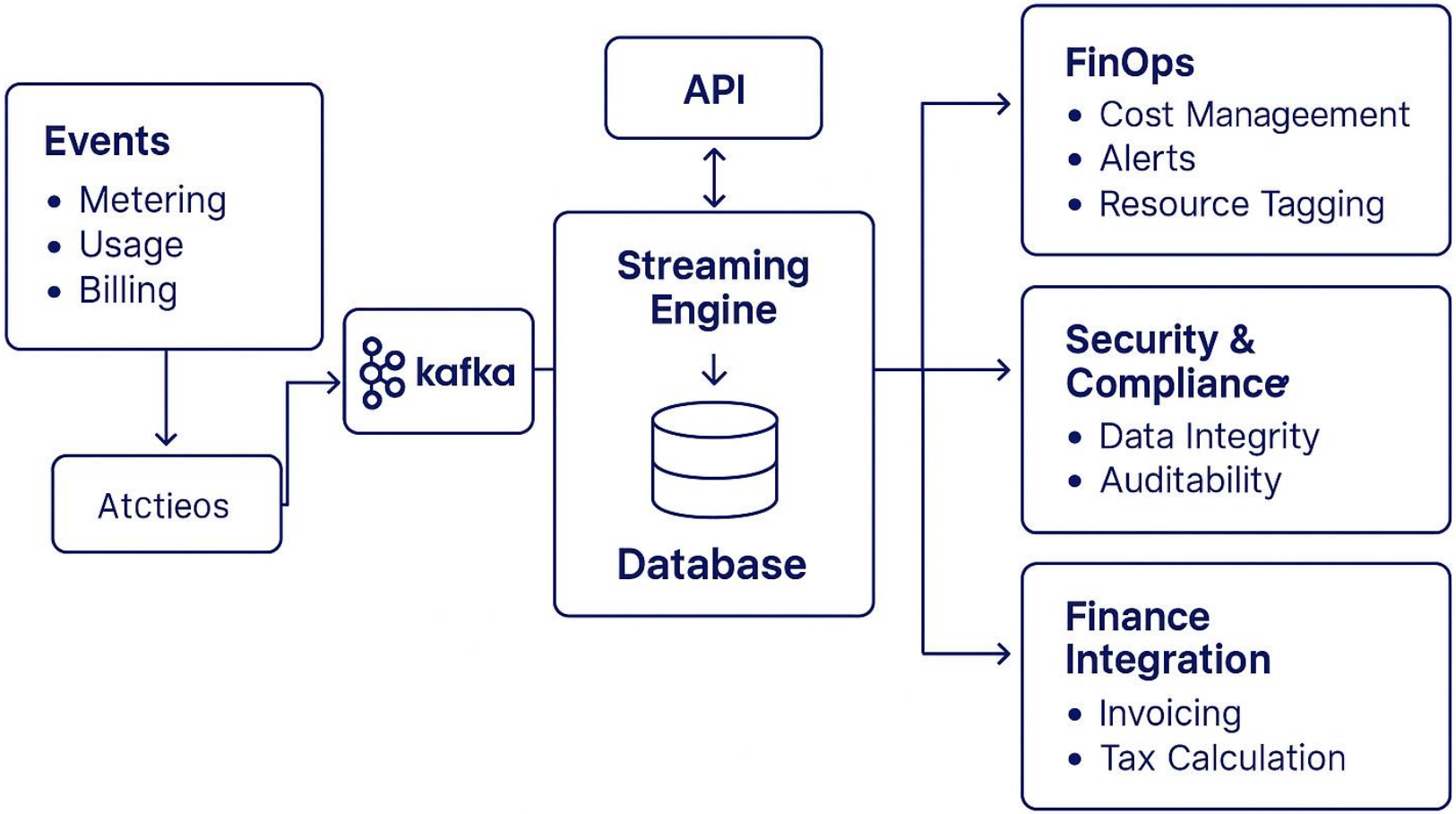
Gamme	vCPU	RAM	Stockage	Bande passante	CPU Type	Usage recommandé
Development	2–48 (partagés)	2–256 GiB	Local ou Block	200–500 Mbps	EPYC 7281 / 7410P (x86)	Dev/test, CMS, microservices
PLAY2 / PRO2	1–32 (partagés)	2–128 GiB	Block	100 Mbps – 6 Gbps	EPYC 7543 (x86)	Web, bases moyennes, coûts optimisés
POP2	2–96 (dédiés)	8–384 GiB	Block	jusqu'à 20 Gbps	EPYC 7543 (x86)	Production, trafic élevé
Compute Optimized	2–96 (dédiés)	4–192 GiB	Block	jusqu'à 10 Gbps	EPYC 7543 (x86)	Calcul intensif, ML, CI/CD
Memory Optimized	2–64 (dédiés)	16–512 GiB	Block	jusqu'à 10 Gbps	EPYC 7543 (x86)	Big Data, cache, base de données in-memory
Specialized (POP2-HM, HC, HN)	2–64 (dédiés)	4–512 GiB	Block	jusqu'à 12,8 Gbps	EPYC 7543 (x86)	RAM, CPU ou réseau optimisés selon besoin

Comparaison avec AWS EC2 (en bref)

- **PLAY2/PRO2** ≈ les familles de type `m5`, `t3` chez AWS (usage général).
- **POP2** (CPU dédiés, up to 96 cores) ≈ EC2 `c5d` ou `m5d` avec CPU garantis.
- **Compute Optimized** ≈ `c5`, `c6i` (haute CPU).
- **Memory Optimized** ≈ `r5`, `r6i` (large mémoire).
- **Specialized (POP2-HM, HC, HN)** ≈ `x1e`, `z1d`, ou familles `r6g` selon ratio RAM/CPU ou réseau.

CORE BILLING

Core Billing



🚀 1. Qu'est-ce que le Core Billing dans un cloud provider ?

Chez Scaleway (comme chez AWS, GCP, OVH), le Core Billing est la **colonne vertébrale de la facturation.**

C'est l'ensemble des services et micro-services qui :

- **Mesurent** l'usage (metering) des ressources consommées (CPU, RAM, stockage, bande passante, requêtes API, egress, etc.).
- **Transforment** ces mesures en **unités billables** (heure de VM, Go stocké par mois, million de requêtes, etc.).
- **Appliquent** la tarification (pricing engine) → souvent des règles complexes : gratuités (free tier), tarifs dégressifs, burst, forfaits mensuels, crédits startup…
- **Génèrent** les factures → en intégrant TVA, devises, comptabilité multi-pays, contraintes légales (RGPD, directive européenne, etc.).
- **Exposent** l'information au client via le portail et l'API (dashboards de consommation en temps réel).

👉 C'est critique car un bug dans le billing = perte de revenus directe ou litige client.

a) Metering & Usage tracking

- Collecte en temps réel ou batch des métriques d'utilisation.
- Exemple : une instance DEV1-S consomme **2 vCPU + 2 GiB RAM + 20 GB block storage** → traduits en "x heures vCPU, y Go.h stockage".
- Outils internes : souvent basé sur **Kafka, Prometheus, ClickHouse ou BigQuery-like** pour stocker les events.

b) Pricing Engine

- Application de règles de pricing :
  - À l'heure (instances Compute).
  - Au volume (Object Storage : €/Go/mois).
  - À la requête (Functions serverless).
  - Forfait mensuel (Web hosting).
- Gestion des promotions, crédits startup (jusqu'à 36 000 € offerts), free-tiers (75 GB egress gratuit).

c) Billing & Invoicing

- Génération des factures PDF légales.
- Intégration avec ERP/CRM (ex. SAP, Netsuite).
- Gestion multi-devise, multi-TVA (client UE, hors UE).

d) Customer Portal & API

- Dashboard temps réel (conso + prévision fin de mois).
- API Billing (permet de récupérer les coûts programmatiquement).
- Gestion des **alertes de consommation** (quota, plafond).

📊 3. Scaleway Billing vs AWS/GCP

- AWS = **Cost Explorer + Budgets + CUR (Cost & Usage Reports)** → très complet mais complexe.
- Scaleway = volonté de rester **transparent et simple**, tarification claire (pas d'effet "surprise bill" comme AWS).
- Exemple concret :
  - Scaleway **Compute Optimized instance C2-M** : 2 vCPU dédiés + 8 GiB RAM = **0,076 €/h** (≈ 55 €/mois).
  - AWS EC2 c6i.large (2 vCPU, 4 GiB RAM) ≈ **0,085 $/h** (≈ 62 €/mois).

    👉 Scaleway plus compétitif en entrée de gamme, mais AWS a plus de granularité (Spot, Savings Plans).

🛠️ 4. Points sur lesquels on peut te challenger en entretien

- **Architecture technique d'un Core Billing** → comment tu designes une plateforme de facturation temps réel, scalable, tolérante aux pannes ?
- **Exemple concret** : une instance est lancée à 10h12 et arrêtée à 11h47 → comment tu calcules le coût exact (prorata à la seconde ou arrondi à l'heure) ?
- **Optimisation de coûts** : comment aider un client à mieux contrôler sa facture (alertes, cost explorer, tagging des ressources).
- **Sécurité et conformité** : garantir l'intégrité des données de facturation (pas de perte d'event metering).
- **Intégration finance** : lien entre usage cloud (events techniques) et écriture comptable/TVA (ex. facturation France vs Allemagne).

AGENDA - BRIQUES ESSENTIELLES

- 1 Architecture technique d’un Core Billing
- 2 Exemple concret de calcul de coût
- 3 Optimisation de coûts (vision client)
- 4 Sécurité & conformité
- 5 Intégration finance & TVA

Architecture technique d’un Core Billing

- a) Collecte (Metering Layer)
- b) Processing (Usage Aggregator)
- c) Pricing Engine
- d) Storage & Billing DB
- e) API & Dashboard
- f) Résilience

a) Collecte (Metering Layer)

- Chaque ressource cloud (VM, bucket, DB, Kubernetes pod, fonction serverless) génère des **events d'usage** (start, stop, requête API, volume transféré).
- Ces events sont envoyés dans un **bus de messages distribué** (Kafka, Pulsar, NATS).
- Format typique :

```json
{
  "resource_id": "vm-12345",
  "event_type": "start",
  "timestamp": "2025-09-01T10:12:00Z",
  "parameters": { "cpu": 2, "ram": 8, "zone": "fr-par-1" }
}
```

b) Processing (Usage Aggregator)

- Un moteur d'agrégation (ex : **Flink, Spark Streaming, Beam**) calcule la durée d'utilisation et les volumes consommés.
- Données stockées dans une **base time-series** (ClickHouse, InfluxDB, BigQuery-like).

c) Pricing Engine

- Les métriques sont traduites en **unités billables** (vCPU.h, Go.h, requêtes).
- Application de règles tarifaires (on-demand, forfait, gratuité, palier dégressif).
- Le moteur doit être **idempotent** : recalculer à partir des events bruts sans incohérence.

d) Storage & Billing DB

- Les données agrégées + factures sont stockées dans une DB relationnelle (PostgreSQL, Aurora, CockroachDB) pour fiabilité et intégrité.

e) API & Dashboard

- REST/GraphQL API expose la consommation en temps réel.
- Interface client (console cloud) → "Cost Explorer maison".

f) Résilience

- **Idempotence** : chaque event peut être retraité sans créer de doub
- **Tolérance aux pannes** : persistance des events dans Kafka + relect
- **Scalabilité horizontale** : workers de traitement parallèles.

Architecture technique d’un Core Billing

- a) Collecte (Metering Layer)
- b) Processing (Usage Aggregator)
- c) Pricing Engine
- d) Storage & Billing DB
- e) API & Dashboard
- f) Résilience

2️⃣ Exemple concret de calcul de coût

Cas : instance lancée à 10h12 et arrêtée à 11h47.

- **Collecte** :
  - Event `START` : 10:12
  - Event `STOP` : 11:47
- **Durée brute** : 1h35min → 95 minutes.
- **Facturation** : dépend de la règle choisie par Scaleway :
  - **Prorata à la seconde** → 95 × 60 = 5 700 secondes. Si 0,00002 €/seconde → 0,114 €.
  - **Prorata à la minute** → 95 minutes × 0,0012 €/minute → 0,114 €.
  - **Arrondi à l'heure** (comme certains clouds historiques) → 2h × 0,072 €/h = 0,144 €.

👉 Scaleway annonce en général une facturation **à la seconde (avec 1 minute minimum)** → plus transparent que AWS (qui est souvent au pas de la seconde aussi depuis 2017, mais pas pour toutes les ressources).

3 Optimisation de coûts (vision client)

Un Core Billing moderne doit **aider le client à ne pas avoir de surprise**.

a) Alertes & Quotas

- Alertes e-mail/SMS/Slack si dépassement d'un seuil (ex : +100 € dans le mois).
- Quotas bloquants (stopper les VM si dépassement crédit).

b) Cost Explorer

- Interface avec graphes par service / projet / période.
- Ex : "Instances Compute = 65% du budget, Object Storage = 20%".

c) Tagging & Projects

- Permet de tagger les ressources (projet="marketing", client="Airbus").
- Facturation ventilée par **projet/équipe/client** → répercussion interne possible (refacturation interne / FinOps).

d) Prévisions & simulation

- Calcul du **coût prévisionnel fin de mois** en extrapolant la consommation courante.
- Simulation de coûts : "si je passe de 4 vCPU à 8 vCPU, quel impact ?".

4 Sécurité & conformité

Facturation = argent → donc **sécurité et auditabilité critiques.**

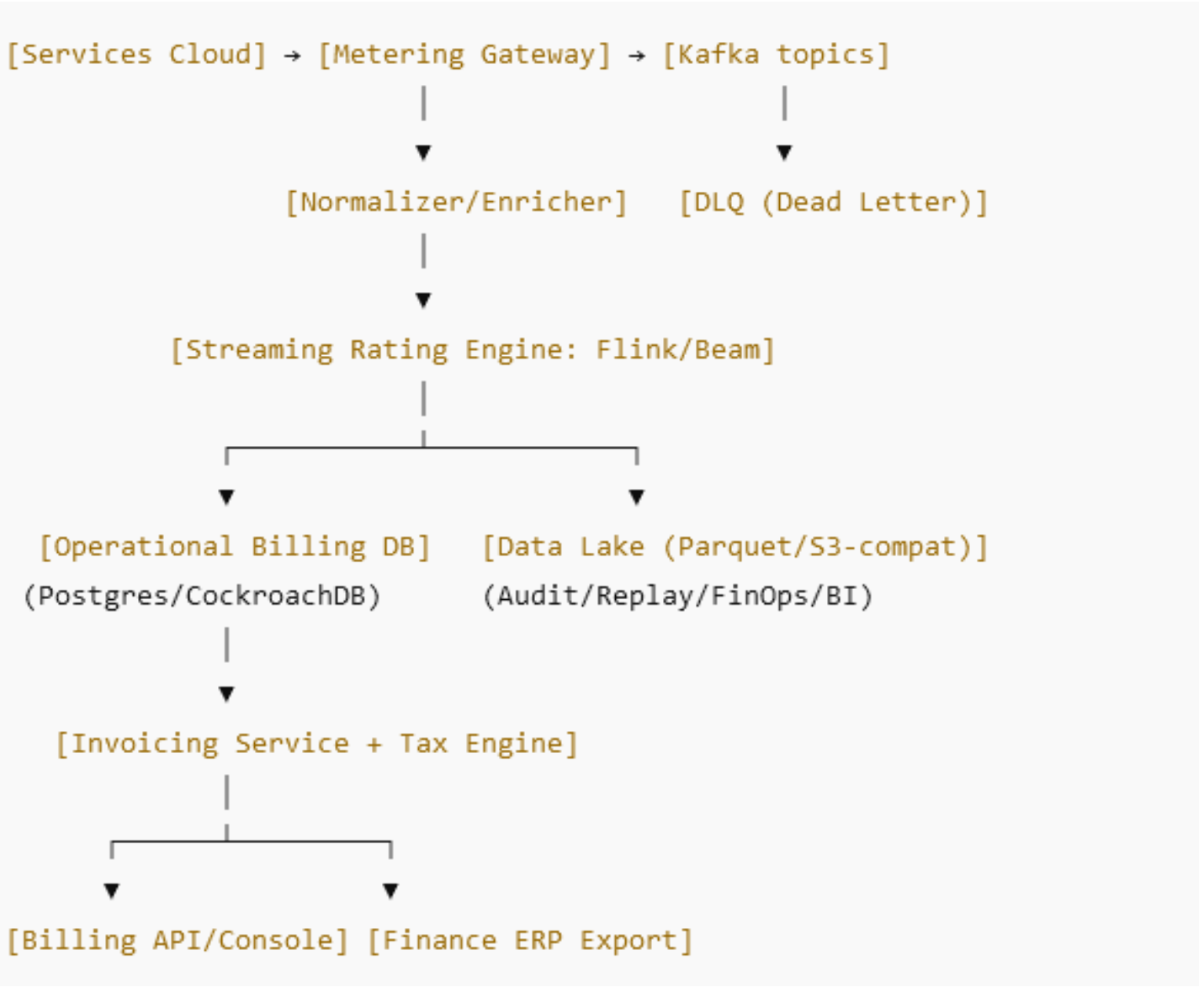
- **Intégrité des données** :
  - Tous les events signés (hash, checksum).
  - Pas de modification a posteriori sans traçabilité (WORM – write once, read many).
- **Conformité légale** :
  - Conservation des factures **10 ans** (Code de commerce).
  - RGPD : anonymisation des logs techniques vs données clients.
- **Auditabilité** :
  - Journaux inviolables (append-only log).
  - Vérification croisée usage ↔ facture → pas de "perte d'événement".
- **Résilience** :
  - Stockage redondant (multi-AZ).
  - Backup quotidien/hebdo de la base de facturation.

5 Intégration finance & TVA

Un système de billing cloud ne se limite pas à la technique : il doit **s'intégrer avec la finance et la législation**.

- **Multi-devise** : facturer en € (Europe), $ (US), PLN (Pologne).
- **Multi-TVA** :
  - Client français : TVA 20%.
  - Client entreprise UE avec numéro de TVA valide : facturation HT (autoliquidation).
  - Client hors UE : facturation HT (règles export).
- **Intégration ERP** : export vers outils financiers (SAP, Netsuite, Sage).
- **Consolidation** : gérer des factures par projet, ou une facture globale consolidée par client.
- **Crédits & remises** : appliquer correctement les coupons (ex : Startup Program Scaleway = 36 000 € de crédits).

MISE EN OEUVRE


[Services Cloud] → [Metering Gateway] → [Kafka topics]
[Normalizer/Enricher]
[DLQ (Dead Letter)]
[Streaming Rating Engine: Flink/Beam]
[Operational Billing DB]
(Postgres/CockroachDB)
[Data Lake (Parquet/S3-compat)]
(Audit/Replay/FinOps/BI)
[Invoicing Service + Tax Engine]
[Billing API/Console]
[Finance ERP Export]

AGENDA MISE EN OEUVRE CORE BILLING

- 1) Architecture d’une plateforme Core Billing (temps réel, scalable, tolérante aux pannes)
- 2) Calcul d’un coût exact au prorata (exemple chiffré)
- 3) FinOps intégré : éviter les “surprise bills”
- 4) Sécurité & conformité (intégrité, audit, RGPD)
- 5) Intégration Finance & TVA (UE)

Couches & responsabilités

- **Metering Gateway**
  - Ingère *tout* événement d'usage (VM start/stop, volume attach/detach, octets sortants, requêtes API, IOPS, etc.).
  - Normalise les timestamps (UTC), ajoute un **idempotency_key** (ex. hash(resource_id+ts+etype)).
  - Push dans **Kafka** (ex. `metering.raw.v1`) via TLS/SASL.
- **Normalizer / Enricher**
  - Convertit unités (MiB→GiB), joint **catalogue des produits** (SKU) et **contexte tarifaire** (région, projet, tags).
  - Écrit dans `metering.normalized.v1`.
  - Toute ligne invalide → **DLQ** (diagnostic, reprocess).
- **Streaming Rating Engine (Flink/Beam)**
  - Logique **stateful** par `resource_id` (fenêtrage start/stop, compteurs cumulés).
  - Gère **late events** (watermarks), **exactly-once** (idempotence) et **re-rating** (si prix/taxe changent rétroactivement).
  - Produit des "**rated usage lines**" (coût élémentaire) vers `billing.rated.v1`.
- **Operational Billing DB** (PostgreSQL/CockroachDB)
  - Tables transactionnelles pour **usage agrégé**, **crédits**, **budgets/alertes**, **invoices**.
  - Contraintes d'intégrité, numérotation légale, historisation **WORM** des factures.
- **Data Lake** (Object Storage S3-compatible, Parquet)
  - **Vérité immuable** des événements bruts/normalisés + rated lines pour **audit/replay** et FinOps/BI (ClickHouse/Trino/BigQuery-like).
- **Invoicing & Tax Engine**
  - Agrège par compte/période, applique **tax rules**, devises, remises, crédits.
  - Gèle le FX rate à l'émission (ECB du jour T-1), calcule la TVA (B2B UE, B2C, hors UE).

2) Calcul d'un coût exact au prorata (exemple chiffré)

Cas : Instance on-demand créée à **10:12**, arrêtée à **11:47** (même fuseau).

Le moteur calcule en **UTC** pour cohérence, et présente dans le fuseau du client (ex. Europe/Madrid).

- Durée = 1 h 35 min = **95 minutes** = **5 700 secondes**.
- Règle typique (ex. compute) : **facturation à la seconde** avec **minimum 60 s** au démarrage (la vraie règle dépend du provider, ici c'est un design recommandé).
- Prix horaire (exemple) = **0,072 € / h**.
  - Prix/seconde = 0,072 / 3600 = **0,00002 €**.
  - Coût = 5 700 × 0,00002 = **0,114 €**.
- Arrondi monétaire : **0,11 €** si arrondi au centime (banque = arrondi half-even).

Points durs gérés par le moteur

- **Burst/stop & start** répétés : on ferme la fenêtre précédente et on ouvre la suivante, agrégation sans trous.
- **Late events** : si l'event STOP arrive tard, on réconcilie et **re-note** (re-rating) la ligne d'usage.
- **Changement de prix en cours de mois** : rating **pro-rata par segment temporel** (avant/après).
- **Crédits/remises** : appliqués **après** rating unitaire, **avant** taxes.

3) FinOps intégré : éviter les “surprise bills”

Tagging, Projects, Cost-Centers

- Impose **tags obligatoires** (projet, équipe, client, environnement).
- Hérite tags **org → projet → ressource** ; refuse création si tags requis manquants (optior

Budgets, Alertes, Quotas

- **Budgets mensuels** par projet + seuils (50/80/100%).
- Alertes **email/Slack/webhook** en temps quasi-réel.
- **Quotas** durs (ex. bloque nouvelles VM au-delà d’un plafond) ou **soft** (alerte only).

Cost Explorer (API + UI)

- Découpe par **service/compte/projet/tag/région**.
- Drill-down niveau **ressource**.
- **Prévisions** fin de mois (extrapolation Holt-Winters ou simple prorata).
- Ex. requête analytique (ClickHouse/SQL) :

```sql
SELECT project, service, sum(cost_eur) AS m_cost
FROM rated_usage
WHERE usage_date BETWEEN '2025-09-01' AND '2025-09-30'
GROUP BY project, service
ORDER BY m_cost DESC;
```

4) Sécurité & conformité (intégrité, audit, RGPD)

Intégrité & traçabilité

- **Chaîne immuable** : événements **signés** (HMAC) à l'ingestion ; stockage brut en **Parquet** sur objet avec verrouillage WORM/Legal Hold.
- **Idempotence** : `idempotency_key` + contraintes **UNIQUE** côté DB.
- **Merkle checks** (optionnel) pour lots d'events → preuve d'intégrité.
- **Audit trail** append-only pour toute correction (re-rating, annulation, note de crédit).

Sécurité des flux & données

- TLS partout (Kafka, API), **mTLS** interne, **KMS** pour clés (chiffrement at-rest & in-transit).
- Secrets en **Secret Manager**, rotation automatique.
- Ségrégation des rôles (RBAC), accès **read-only** pour FinOps/Finance si besoin.

Conformité & rétention

- **RGPD** : minimisation (pas de données perso inutiles dans les events), base légale (contrat), **DPIA** pour le système.
- **Rétention** : événements bruts N années (audit), rated lines ≥ 10 ans (exigences comptables), logs accès ≤ 6-12 mois selon politique.
- **Factures** : PDF signé, archive WORM, horodatage qualifié (selon législation).

5) Intégration Finance & TVA (UE)

Modèle comptable

- Chaque **rated line** (HT) → **écriture comptable** (GL) avec **contreparties** :
  - Produits (par SKU / compte comptable),
  - Clients (comptes auxiliaires),
  - Taxes (TVA collectée).
- **Devise de tenue** = EUR ; **devise client** affichée si besoin.
- **Taux de change** figé à l'émission (source BCE), stocké avec référence (date/source).

Règles TVA (UE — simplifiées)

- **B2B UE** (client avec n° TVA valide VIES) → **autoliquidation** (facture HT, mention légale).
- **B2C UE** → TVA du pays du client (guichet OSS si appliqué).
- **Hors UE** → HT (export), attention à la **place of supply** des services numériques.
- **France** → TVA FR 20% par défaut, cas particuliers DOM/TOM.
- Le **Tax Engine** choisit la règle selon : *pays de facturation*, *type client*, *preuve de localisation*.

Cycle de facturation & corrections

- **Pré-facture** (proforma) → validations, application crédits/remises.
- **Facture finale** → numérotation séquentielle, PDF scellé, envoi + export ERP (SAP/Netsuite).
- **Avoirs** : re-rating tardif → **note de crédit** rattachée à la facture d'origine (audit trail).

CONCLUSIONS

POINTS IMPORTANTS A PRENDRE EN COMPTE

- **Exactly-once** logique via idempotence + replay ; **late events** gérés (watermarks) ; **re-rating** sûr.
- **Facturation au prorata** (seconde/minute) avec **min charge**, multi-segments si prix change.
- **FinOps by design** : tagging obligatoire, budgets/alertes/quotas, explorer & prévisions.
- **Sécurité/audit** : WORM, signatures, KMS, journal inviolable, conformité RGPD & rétention légale.
- **Finance/TVA** : règles UE (B2B autoliquidation, B2C OSS), FX BCE, export ERP, écritures double-partie.