NECKO technologies

L'utilisation des agents IA et des serveurs MCP dans un contexte client

aws PARTNER Advanced Tier Services

Aperçu de la présentation

1. Présentation de Necko
2. Présentation des agents IA et des MCP
3. Introduction au Cloud AWS et au Serverless
4. Présentation de l'IA sur AWS
5. Démonstration - Compose ton PAE avec un agent
6. Conclusion et contacts

Présentation de Necko

Nous accompagnons les entreprises dans leur voyage vers le cloud

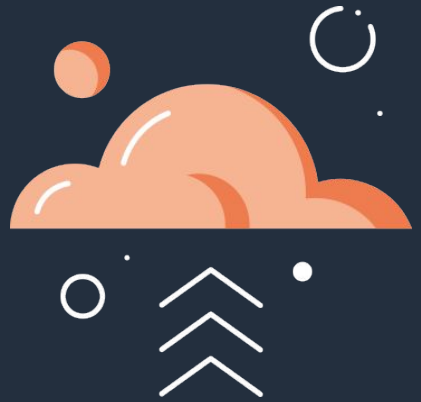
Nous leur offrons:

- une expertise pointue sur les services et les produits AWS (mais aussi GCP et Azure)
- du développement software
- des services de consultance

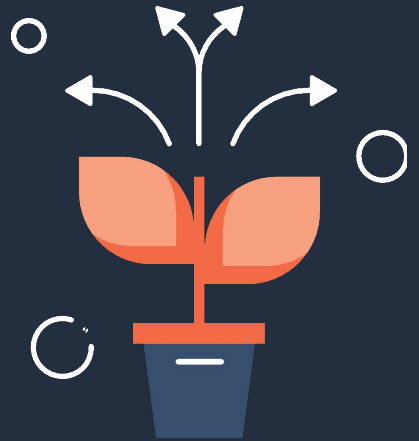
NECKO technologies

Notre offre de services

Foundation



Migration



Optimisation

NECKO technologies

En quelques mots

aws PARTNER Advanced Tier Services

- AWS Lambda Delivery
- AWS CloudFormation Delivery
- Well-Architected Partner Program

- Startup fondée en 2017
- Basée à Mons
- Partenaire cloud AWS depuis 2017, premier partenaire wallon du cloud AWS
- AWS Advanced Partner, unique en Wallonie
- AWS WAR Partner
- Expertise spécifique AWS

Prix Mercure 2018
Ville de Mons
Jeune Entreprise

lme la maison de l'entreprise

Inno pépites 2018
LME
Catégorie entreprise IT

AWS Service Delivery
AWS Lambda
AWS CloudFormation

NECKO technologies

WAR: Well Architected Framework

L’équipe

- 18 collaborateurs (pour le moment… On recrute 😁)
- Profils architectes et développeurs
- Plus de 100 certifications AWS

NECKO technologies

Nos clients


EQUANS


VERBOLIA


LIZY


AGC


TelSmart


IPM
GROUP
Media Today


ENGIE


n-hertz


TRACTEBEL


ENGIE


KASPARD


INGESTIC
SMART DATA MINDED COMPANY


NECKO
technologies

Agentic AI

Les Agents IA

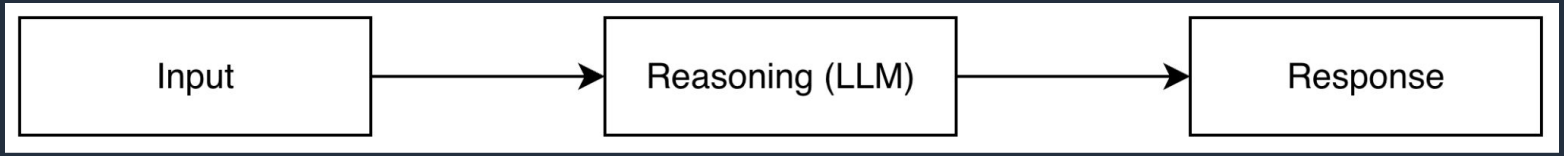
“A language model can answer questions.

An agent can do things.

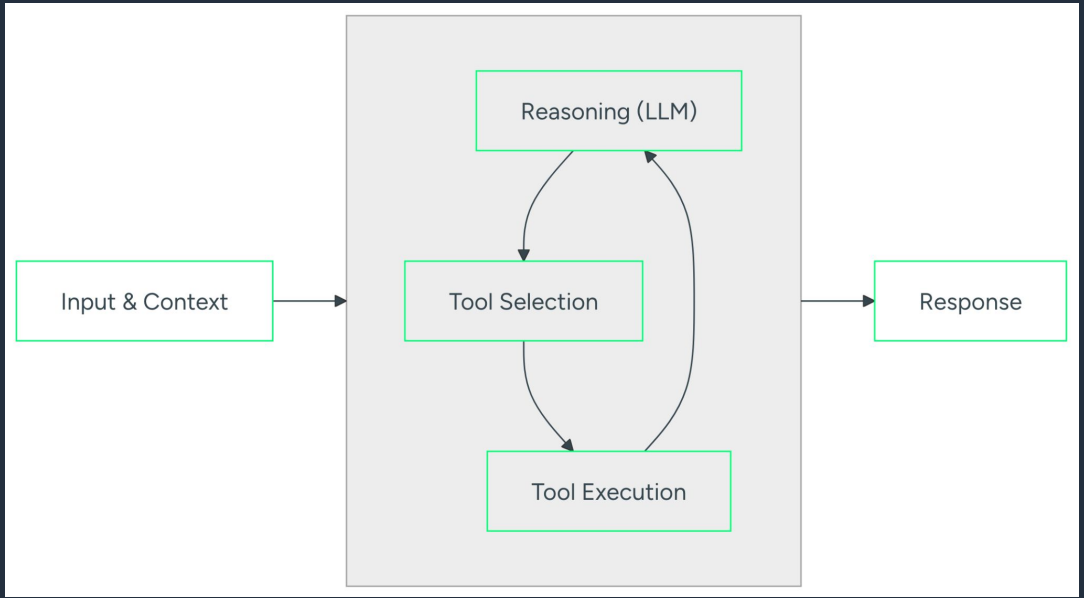
The agent loop is what makes that difference possible”

- Strands Agents Documentation

Les Agents IA



Les Agents IA



Les Agents IA en entreprise

Ce qui change pour les entreprises

Avant	Avec l'agentic AI
- Tâches simples - Le LLM répond, l'humain agit	- Tâches complexes et contextuelles - Humain en superviseur, agent en exécution - Intégration d'**outils**

Les Agents IA

Les composants clés :

- Le LLM - moteur de raisonnement
  - Décide quelle action prendre à chaque étape

- La mémoire
  - In-Context : Ce qui est dans le prompt courant
  - Externe : Vector store (RAG)
  - Éphémère : Résultat des étapes précédentes

Les Agents IA

- Les outils
  - Fonctions que l'agent peut appeler : API, database, calculatrice, etc.
  - C'est ce qui donne à l'agent un effet sur le monde réel

- Les patterns de raisonnement
  - ReAct (reason and act)
  - CoT (chain of thought)
  - ReWOO (reason without observation)

Agents simples VS Multi-agents

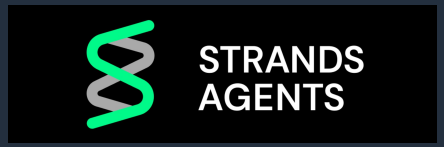
Agent simple :

- Un LLM + ensemble d'outils + une boucle
- Adapté pour des tâches ciblées et bien définies

Multi-agents :

- Des agents individuels qui collaborent
- Des LLM et outils différents pour chaque sous-agent

Frameworks



Et plein d'autres…

Strands Agents

- Open source SDK
- Léger
- Model agnostic
- Model-driven
- Intégrations avec des **outils**

Strands Agents

```python
from strands import Agent
from strands_tools import calculator

agent = Agent(tools=[calculator])
response = agent("Combien fait 1337 * 42 ?")
```

Les outils (Tools)

Définition

- Mécanisme élémentaire pour étendre les capacité d'un agent
- Une fonction qu'il peut décider d'appeler pour interagir avec le monde extérieur

Les outils (Tools)

Un outil est composé de trois éléments :

- Nom : identifiant unique
- Description : Texte en langage naturel
- Schéma : structure des paramètres d'entrée/sortie (en JSON)

Les outils (Tools)

Comment le LLM “choisit” un outil:

- Le LLM ne voit pas le code de l’outil
- Il voit uniquement le nom, la description en texte et le schéma
- Il génère une utilisation de tool de la forme:

```
{
  tool: search_products,
  parameters: {
    query: chaise ergonomique,
    limit: 5
  }
}
```

Les outils (Tools)

```python
from strands import tool


@tool
def get_order_status(order_id: str) -> dict:
    """
    Retourne le statut d'une commande client.
    Utilise cet outil quand l'utilisateur demande où en est sa commande.

    Args:
        order_id: L'identifiant unique de la commande (format ORD-XXXX)
    """
    return db.query(f"SELECT status FROM orders WHERE id = '{order_id}'")
```

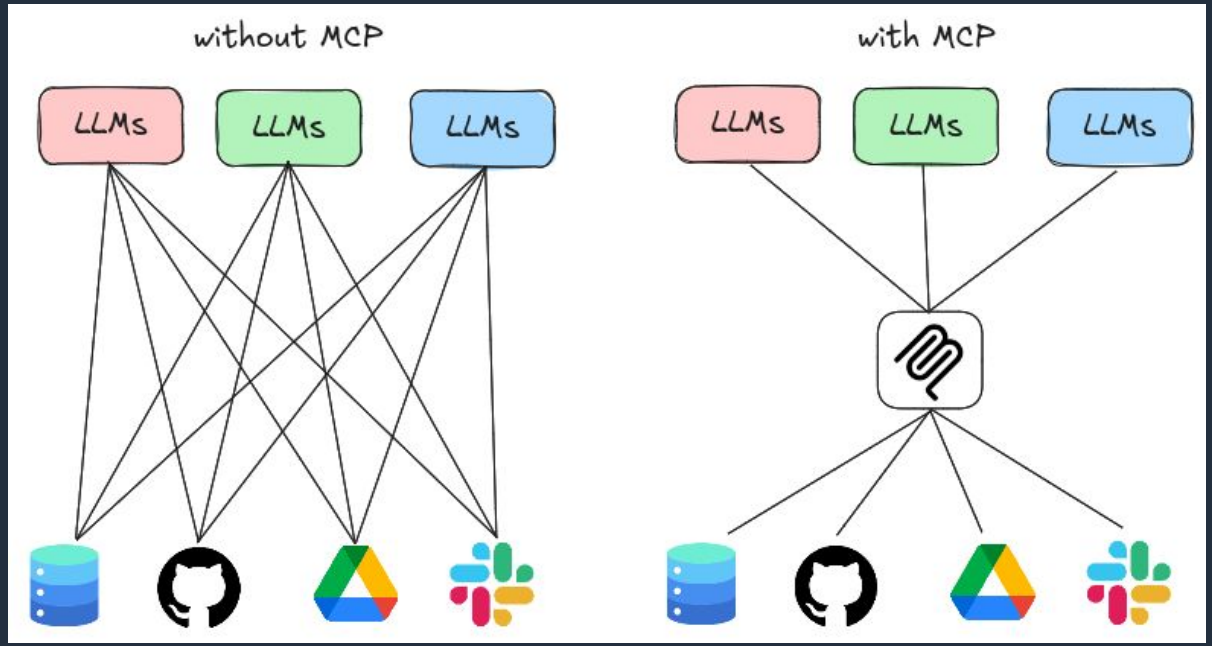
Les outils (Tools)

Avant MCP :

- multiplication des frameworks agent
- multiplication des tools

-> connecteur custom pour chaque framework et chaque tools

Les outils (Tools)



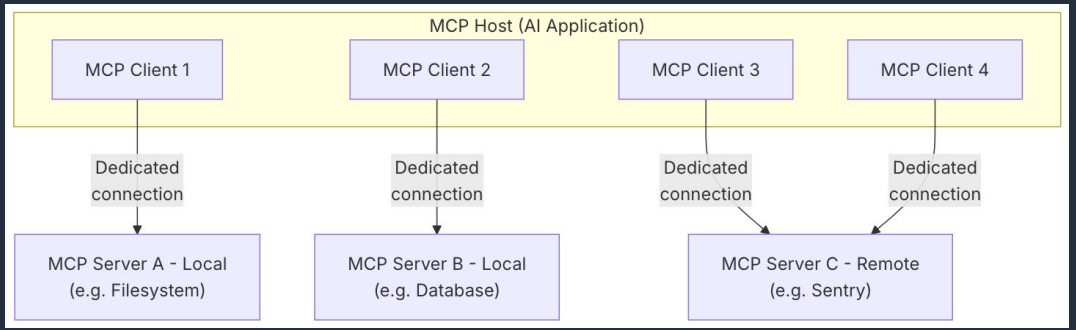
Model Context Protocol

- Introduit par Anthropic en 2024
- Protocole ouvert qui standardise l’accès aux outils

MCP - Architecture

3 acteurs :

- Host (application)
- MCP client
- MCP server



MCP - Capacités

3 types de capacités qu'un serveur peut exposer :

- Tools - ce que l'agent peut faire
- Resources - ce que l'agent peut lire
- Prompts - templates réutilisables

MCP - Implémentation

```python
from mcp.server.fastmcp import FastMCP

mcp = FastMCP("Serveur CRM")

@mcp.tool()
def get_customer(customer_id: str) -> dict:
    """
    Retourne les informations d'un client.
    Utilise cet outil pour récupérer le profil d'un client par son ID.
    """
    return crm_db.find(customer_id)

@mcp.resource("crm://customers/list")
def list_customers() -> str:
    """Liste tous les clients actifs"""
    return crm_db.get_all_active()

if __name__ == "__main__":
    mcp.run()  # démarre le serveur en mode stdio par défaut
```

```python
from strands import Agent
from strands.mcp import MCPClient

# Connexion au serveur MCP local
mcp_client = MCPClient(
    transport="stdio",
    command=["python", "crm_server.py"]
)

# Les outils MCP sont automatiquement découverts et injectés
with mcp_client:
    agent = Agent(
        tools=mcp_client.list_tools(),  # get_customer, etc.
        system_prompt="Tu es un assistant CRM..."
    )
    response = agent("Donne moi le profil du client C-1042")
```

MCP - Dans un contexte client

- Permet de centraliser les outils et la connaissance
  - La documentation interne
  - La logique métier

- Permet de changer de provider sans tout réécrire

AWS Cloud

Qu’est ce que le Cloud ?

Le cloud computing est la mise à disposition de ressources informatiques à la demande via Internet, avec une tarification en fonction de votre utilisation.

source: https://aws.amazon.com/fr/what-is-cloud-computing/

NECKO technologies

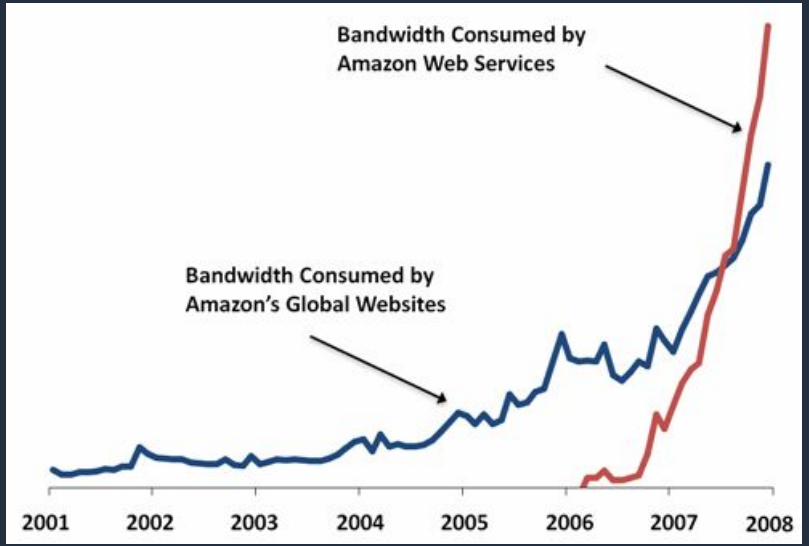
Origine de AWS

Amazon débute avec la vente de livres, puis standardise son infrastructure face à la diversification.

En 2002, ses ingénieurs développent des services réutilisables, aboutissant à S3, SQS et EC2.

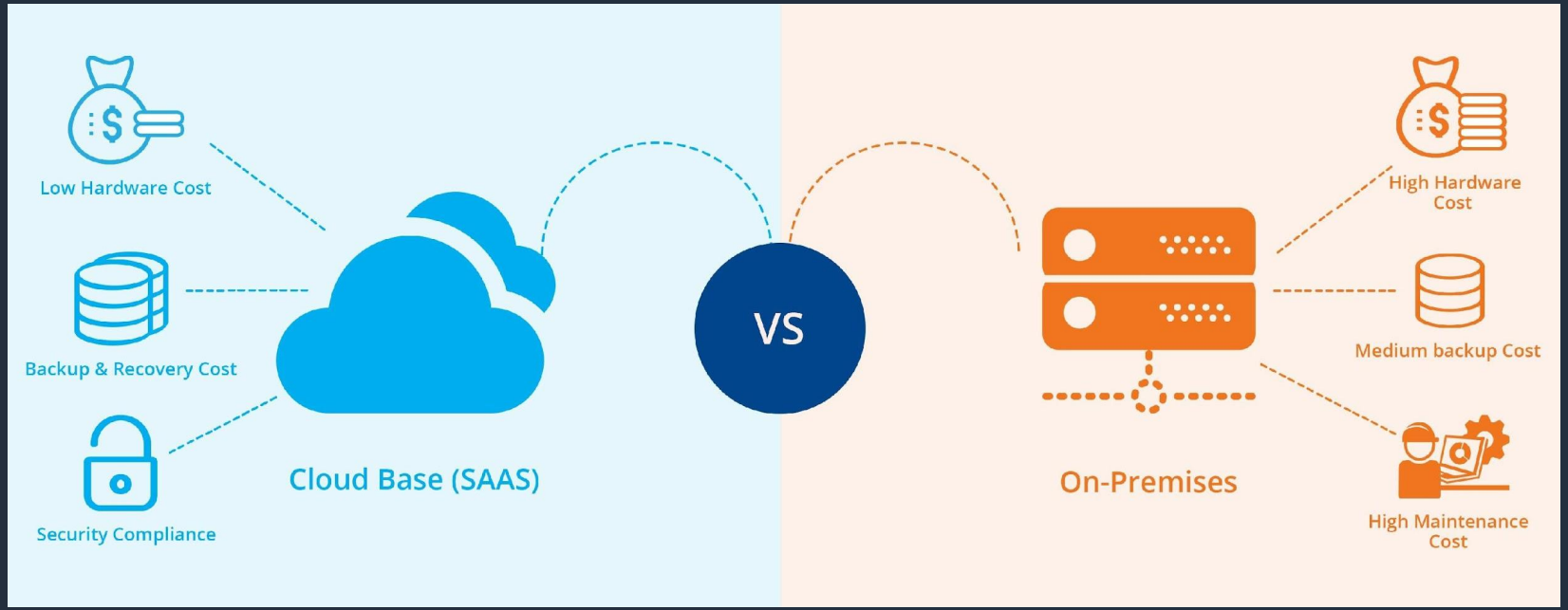
AWS est lancé en 2006 et devient, dès 2010, un leader du cloud.

Depuis 2015, AWS génère plus de bénéfices qu'Amazon Commerce, avec ~20% du CA et 50% à 70% des profits.



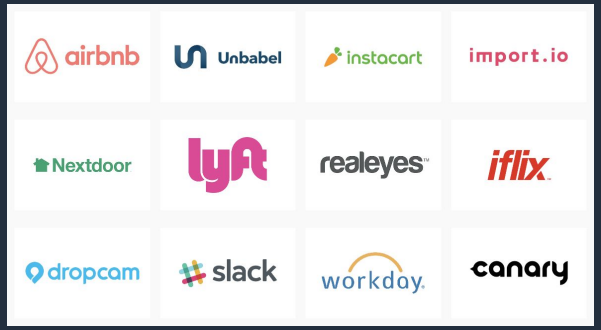
Source:
https://aws.amazon.com/blogs/aws/lots-of-bits/

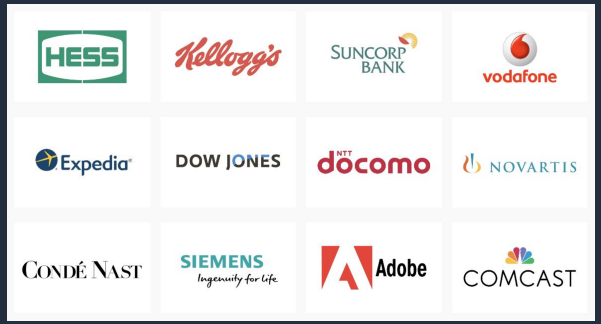
Cloud vs On-Premises



Amazon Web Services

- Pas de frais initiaux
- Paiement à l'utilisation
- Accélération du Time to Market et Agilité
- Scaling automatique
- Infrastructure self-service

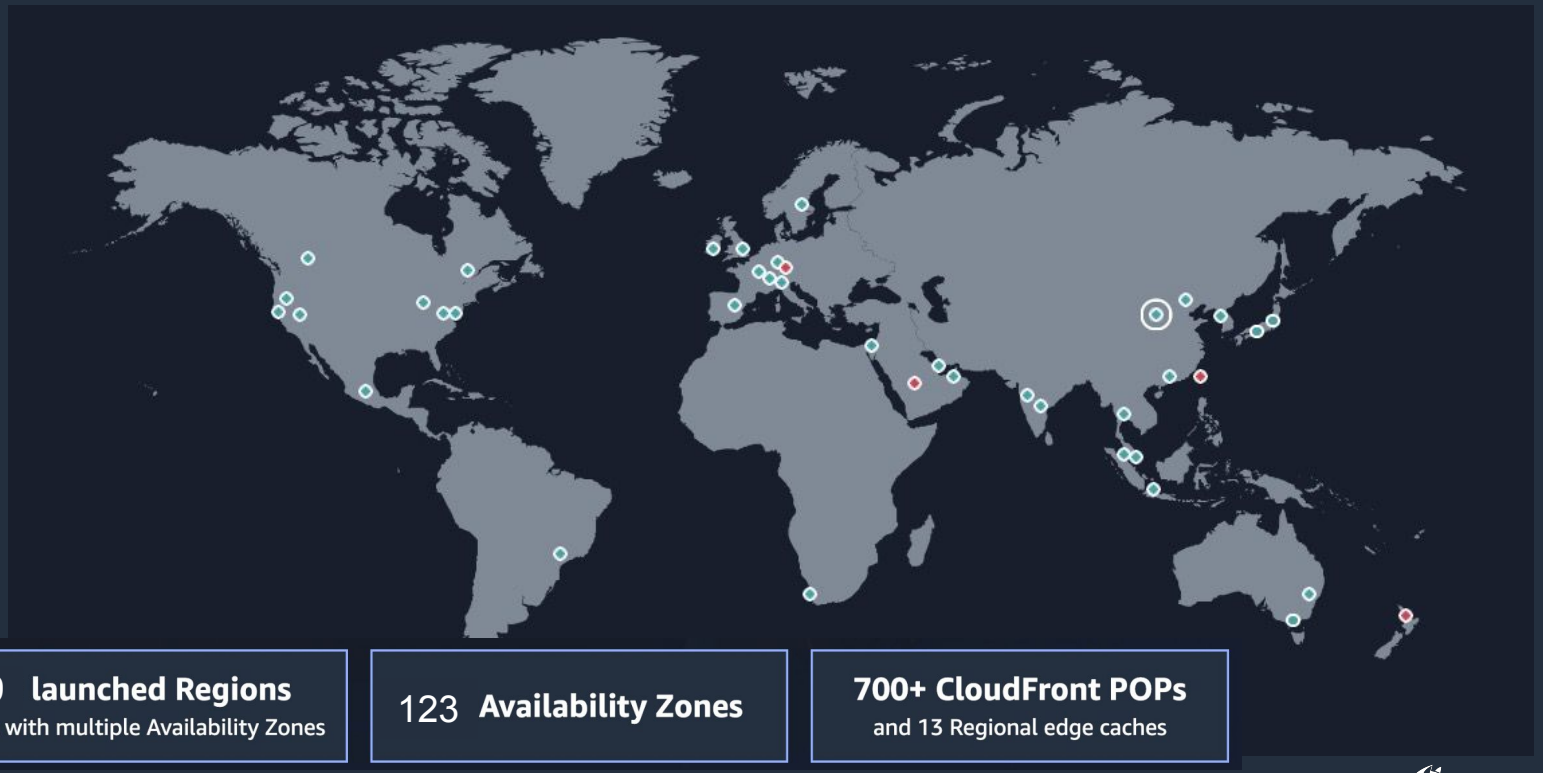




NECKO technologies

AWS leader depuis 15 ans

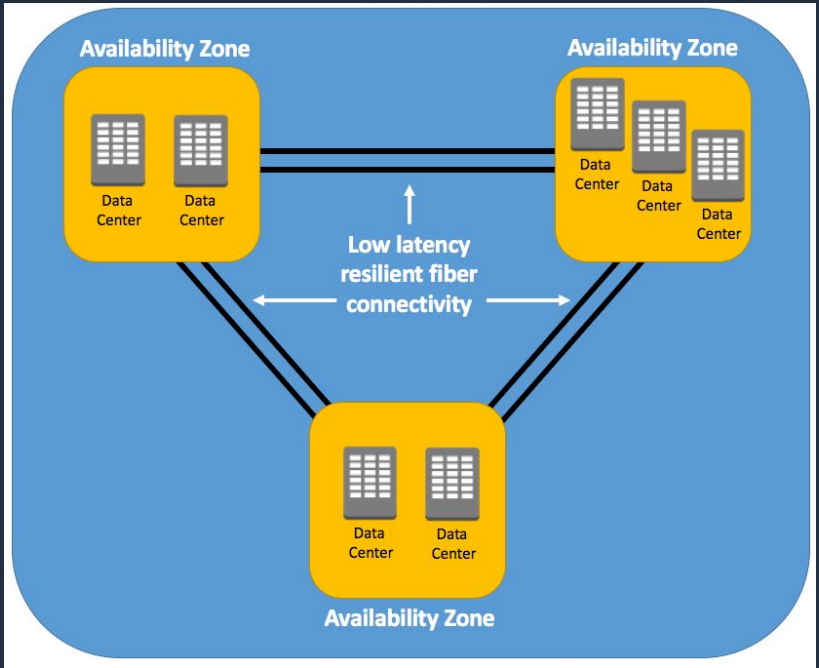
AWS Infrastructure Globale

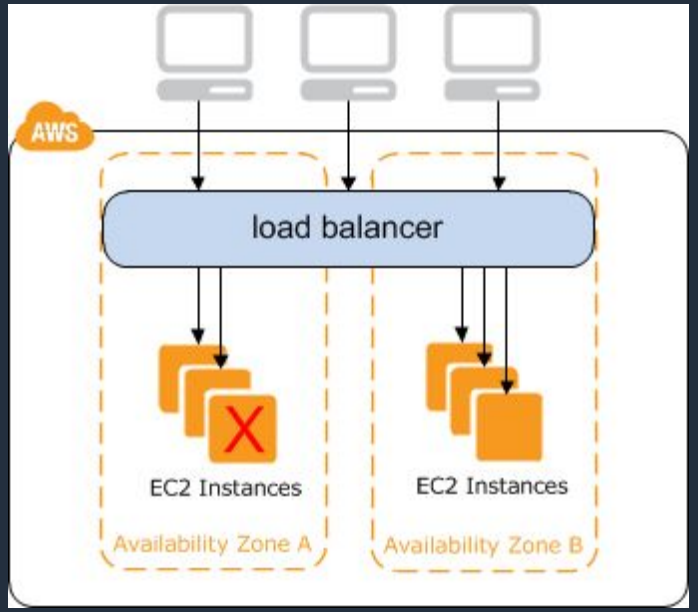


Source : https://aws.amazon.com/about-aws/global-infrastructure/

NECKO technologies

Région divisée en Availability Zone



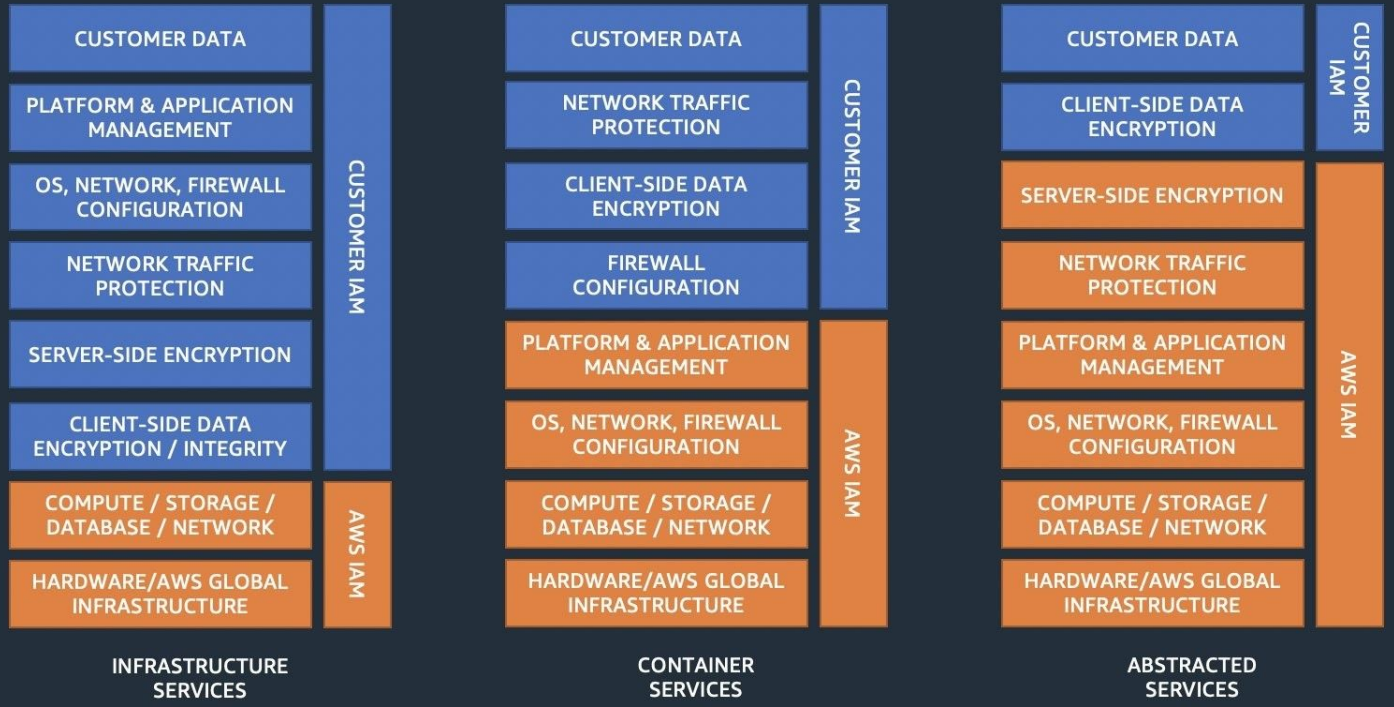


Serveurs virtuels dans le cloud

AWS Elastic
Cloud Compute
(EC2)

- Stockage modulable
- Disponibilité
- Démarrage rapide
- Instance différents
- Payez à l'utilisation

De on-premise vers serverless

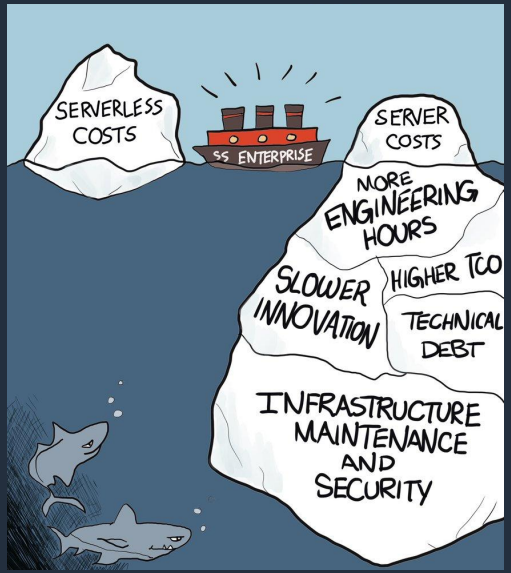


Source:
https://aws.amazon.com/blogs/industries/fsi-service-spotlight-amazon-elastic-container-service-ecs-with-aws-fargate/

NECKO technologies

Managed service = Serverless

- Coût à l'utilisation
- Garantie de fiabilité
- Alerting: une alerte = erreur dans la logique
- Petit services



Autre services AWS

- plus de 200 services
  - **Calcul** : EC2, Lambda, Auto Scaling.
  - **Stockage** : S3 (objet), EBS (bloc), EFS (fichier).
  - **Base de données** : RDS (relationnel), DynamoDB (NoSQL), Aurora.
  - **Réseau** : VPC, Route 53, Direct Connect.
  - **IA & Machine Learning** : SageMaker, Bedrock (IA générative).
  - **Développement** : CodePipeline, CodeCommit

Gestion des identités et des accès

AWS IAM

- Gestion accès AWS
  - Utilisateur
  - Service
- Peut être fédérée avec d'autres systèmes (Microsoft Active Directory)

Database pour documents

Amazon
DynamoDB

- NoSQL
- Interface JSON
- Accès en ms

Database relationnelle serverless

Amazon
Aurora
Serverless V2

- SQL
- Scale to zéro
- Accès en ms
- Petite soeur: Aurora DSQL

S3: Simple Storage Service

Amazon S3

- Stockage de fichiers
- Max 5 Pb/fichier
- Scale (jusqu'à 55000 requêtes/s)

Appel de fonction

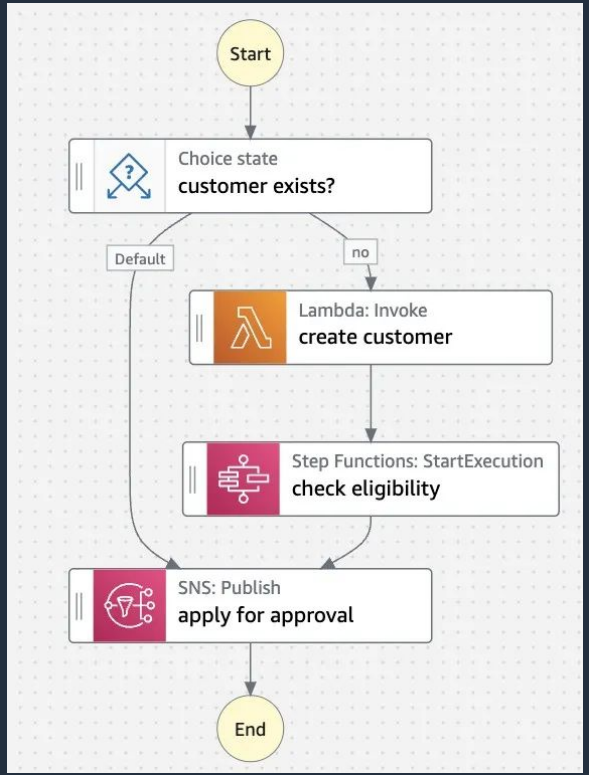
AWS Lambda

- Déclenché par un évènement
  - Cron
  - par action sur S3, Dynamo…
- Scaling automatique
- Exécute du code
- Coût = € x secondes x mémoire alloué

Orchestration de services

AWS Step Functions

- Machine à états
- outil de visualisation de workflow



Infrastructure as Code

Objectif: Décrire l'infrastructure sous forme de code

Avantages:

- Vue détaillée de l'implémentation
- Facile à redéployer la même infrastructure
- Permet de versionner le code (Git)
- Déploiement automatique
- Tests et vérifications automatique

Créer et gérer des ressources à l'aide de modèles

AWS
CloudFormation

- Template au format Yaml
- Rollback en cas de problèmes
- Suivi du *drift*
- Permet de faire un *diff*
- *Managé par AWS*

AWS Cloud Development Kit

```
const table = new Table(this, id: 'table', props: {
  partitionKey: {
    type: AttributeType.STRING,
    name: 'id'
  }
});
const role = new Role(this, id: 'role', props: {
  assumedBy: new ServicePrincipal(ServicePrincipals.LAMBDA)
});
table.grantReadData(role);
```

- Génère du CloudFormation
- Simplifie les liens entre les ressource
- Debuggable
- Permet de créer des packages réutilisables

NECKO technologies

L’IA sur AWS

Vue d’ensemble

Services AWS

- APIs Serverless
  - Rekognition
  - Transcribe
  - Polly
  - Textract
  - …
- Machine learning
  - Sagemaker
- IA Générative
  - AWS Bedrock

IA Générative

AWS Bedrock

- Catalogue de modèles
  - Claude (Anthropic)
  - Mistral (Mistral AI)
  - Llama (Meta)
  - Stable Diffusion (Stability AI)
  - DeepSeek-R1
- API Serverless
- Moderation (Guardrails)
- Outils de Prompt Engineering
- Outils d’évaluation

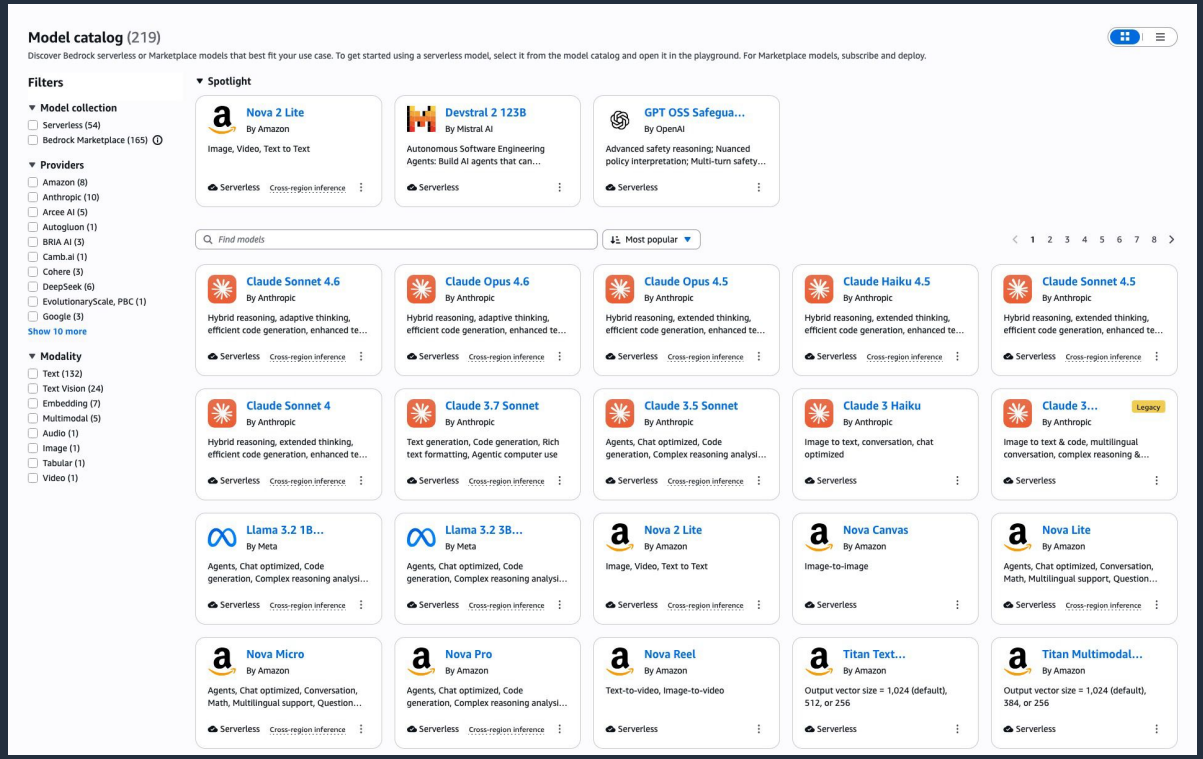
IA Générative

AWS Bedrock

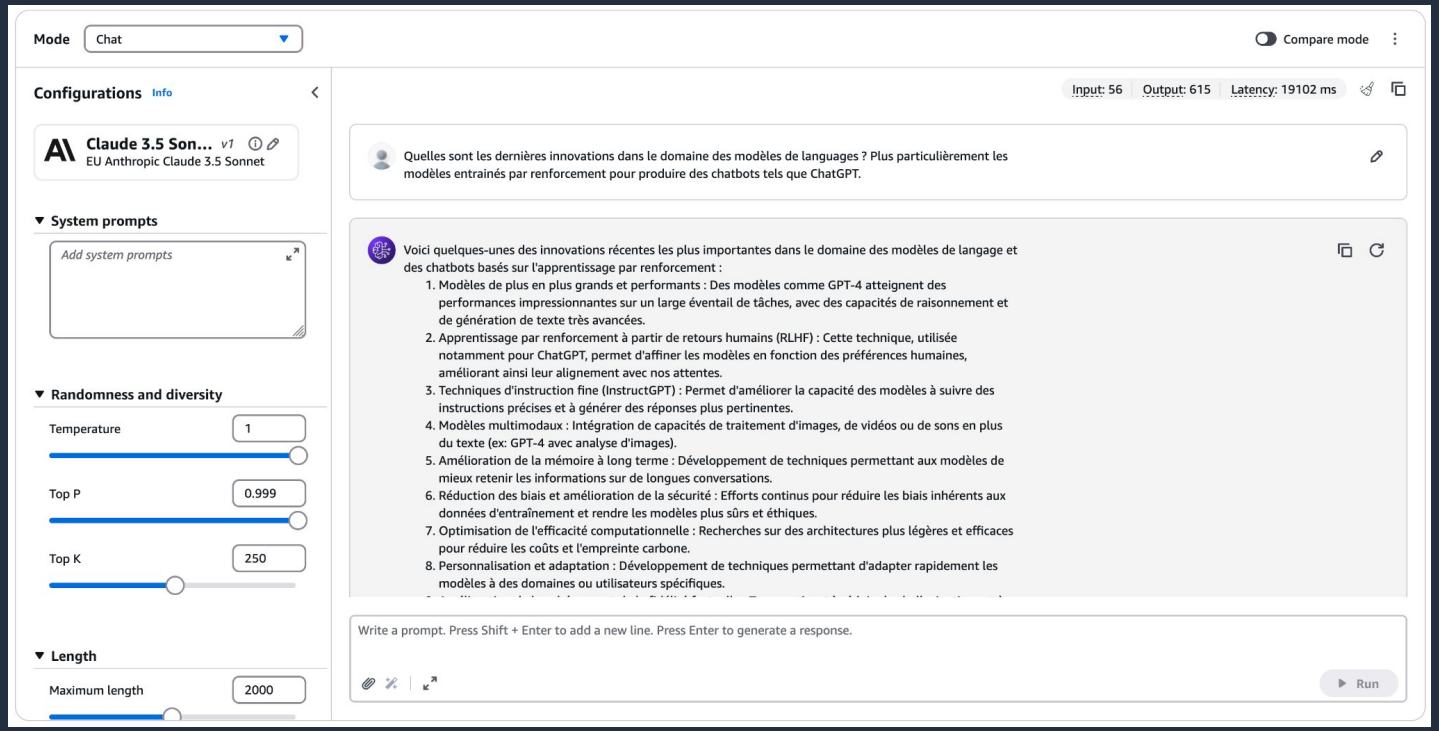
Mais aussi :

- Compréhension d'images / vidéos
- Génération d'images / vidéos
    - Stable Diffusion
    - Amazon Nova
    - Amazon Titan

Catalogue de modèles



Chat


Mode
Chat
Compare mode
Configurations
Info
Claude 3.5 Son... v1
EU Anthropic Claude 3.5 Sonnet
System prompts
Add system prompts
Randomness and diversity
Temperature
1
Top P
0.999
Top K
250
Length
Maximum length
2000
Input: 56
Output: 615
Latency: 19102 ms
Quelles sont les dernières innovations dans le domaine des modèles de languages ? Plus particulièrement les modèles entrainés par renforcement pour produire des chatbots tels que ChatGPT.
Voici quelques-unes des innovations récentes les plus importantes dans le domaine des modèles de langage et des chatbots basés sur l'apprentissage par renforcement :
1. Modèles de plus en plus grands et performants : Des modèles comme GPT-4 atteignent des performances impressionnantes sur un large éventail de tâches, avec des capacités de raisonnement et de génération de texte très avancées.
2. Apprentissage par renforcement à partir de retours humains (RLHF) : Cette technique, utilisée notamment pour ChatGPT, permet d'affiner les modèles en fonction des préférences humaines, améliorant ainsi leur alignement avec nos attentes.
3. Techniques d'instruction fine (InstructGPT) : Permet d'améliorer la capacité des modèles à suivre des instructions précises et à générer des réponses plus pertinentes.
4. Modèles multimodaux : Intégration de capacités de traitement d'images, de vidéos ou de sons en plus du texte (ex: GPT-4 avec analyse d'images).
5. Amélioration de la mémoire à long terme : Développement de techniques permettant aux modèles de mieux retenir les informations sur de longues conversations.
6. Réduction des biais et amélioration de la sécurité : Efforts continus pour réduire les biais inhérents aux données d'entraînement et rendre les modèles plus sûrs et éthiques.
7. Optimisation de l'efficacité computationnelle : Recherches sur des architectures plus légères et efficaces pour réduire les coûts et l'empreinte carbone.
8. Personnalisation et adaptation : Développement de techniques permettant d'adapter rapidement les modèles à des domaines ou utilisateurs spécifiques.
Write a prompt. Press Shift + Enter to add a new line. Press Enter to generate a response.
Run

Bedrock AgentCore

Plateforme d’infrastructure agentique managée par AWS

Permet d’héberger les agents définis dans n’importe quel framework

Bedrock AgentCore - Services

- Runtime

  Déploiement serverless d’agents

- Memory

  Mémoire persistante entre les sessions

- Identity

  Gestion des identités

Bedrock AgentCore - Services

- Gateway

  Connecte les agents aux outils existants / aux MCP

- Policy

  Contrôle des actions autorisées pour chaque agent

- Observability

  Dashboard, logs

Démonstration

Conclusion et contacts

Nous contacter

Necko Technologies

Necko Technologies

Doriano Dell'Aria
Lead Cloud Developer at Necko Technologies

Maximilien Charlier
Solutions Architect at Necko Technologies | PhD | 15x AWS certified | IoT & LoRaWA...

Merci pour votre attention!

Posez vos questions maintenant ou n’hésitez pas à nous contacter !

www.necko.tech

contact@necko.tech

NECKO technologies