

# Agents autonomes et intelligence collective

L'objectif de cette formation est de comprendre les fondements, les architectures, et les outils pour concevoir des systèmes multi-agents intelligents basés sur des LLMs, capables de planifier, raisonner, collaborer, s'adapter et se spécialiser autour de tâches complexes, distribuées ou à forte variabilité contextuelle.

**Pré-requis :** Bonne maîtrise de Python, LLMs et bases LangChain recommandées, architecture logicielle ou orientée agent.



**Modalité :**

- Distanciel en classe virtuelle
- E-learning : à venir
- Présentiel

**Communauté:** [community.reconvert.net](https://community.reconvert.net)

**Durée totale :** 21 H (3 jours)

## PLAN DETAILLE

### Les agents intelligents – principes et terminologie

Définitions

Agent, autonomie, environnement, perception, action, objectifs

Agent symbolique vs LLM-powered

Différence entre agents, chaînes et copilotes

Vision d'ensemble des architectures d'agents

(BDI, planificateurs, prompt-based)

### Introduction aux LLMs dans une architecture multi-agents

Modèle de langage comme moteur de planification et de raisonnement

Utilisation d'outils, mémoire, états internes

Limitations : hallucinations, coordination, coût computationnel

### LangChain Agents et tool-based reasoning

Architecture d'un agent LangChain

Tools, memory, output parser, AgentExecutor

Exemples : calcul, recherche, fichiers, APIs

### Atelier 1 : Création d'un agent simple avec outils

Agent qui répond à des questions, utilise un outil de recherche, fait un calcul

Gestion des chaînes de raisonnement et de l'environnement

### Systèmes multi-agents (MAS) – théories et stratégies

Types d'agents : spécialisés, hiérarchiques, compétitifs, collaboratifs

Approches de coordination : par tâche, par rôle, par message

Modèle de communication : blackboard, publish/subscribe,

dialogue direct (messages JSON)

### Outils et frameworks de conception

LangChain Multi-Agent

CrewAI, AutoGen, ChatDev, AutoGPT

Structuration de workflows complexes : délégation, rôles, dépendances

### Rôle des ontologies et du raisonnement sémantique

Standardisation des connaissances partagées

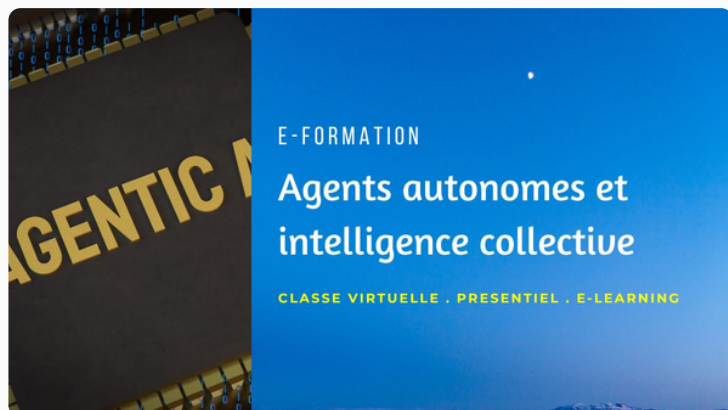
Utilisation d'ontologies métiers dans un système multi-agent

Raisonnement à partir de graphes de connaissances (RDF, Neo4j)

# Agents autonomes et intelligence collective

L'objectif de cette formation est de comprendre les fondements, les architectures, et les outils pour concevoir des systèmes multi-agents intelligents basés sur des LLMs, capables de planifier, raisonner, collaborer, s'adapter et se spécialiser autour de tâches complexes, distribuées ou à forte variabilité contextuelle.

**Pré-requis :** Bonne maîtrise de Python, LLMs et bases LangChain recommandées, architecture logicielle ou orientée agent.



**Modalité :**

- Distanciel en classe virtuelle
- E-learning : à venir
- Présentiel

**Communauté:** [community.reconvert.net](https://community.reconvert.net)

**Durée totale :** 21 H (3 jours)

## PLAN DETAILLE

### Atelier 2 : Conception d'un système à 3 agents spécialisés

Exemples :

- + Un analyste de données
- + Un rédacteur de rapport
- + Un vérificateur juridique

Coordination par rôle et par objectif

Utilisation de mémoire longue

### Planification automatique et reasoning

Chaînes de pensée (chain-of-thought) et planification par LLM

Limitations des LLMs : bruit, instabilité, boucle infinie

Stratégies de contrôle : scoring, pruning, critic agents

### Supervision, monitoring et sécurité

Logging des interactions, replay de dialogue

Gouvernance, sandboxing, auditabilité

Agent critique ou humain dans la boucle (HITL)

### Scalabilité et orchestration

Performance : coût, temps, ressources (CPU, RAM, API)

Orchestration avec Airflow, FastAPI, LangServe

Déploiement d'agents autonomes dans un SI : isolation, scheduling, persistance

### Atelier final : Créer un prototype multi-agent autonome

Création d'un écosystème intelligent (ex. gestion d'un incident IT, pipeline de décision, analyse documentaire)

Documentation, mise sous test, scénario supervisé