Practical Work
Multi-Agent Systems for Smart Home Cooperation

L3 — Cognitive Agents

1 Objectives

The objective of this practical work is to design and implement a **Multi-Agent System (MAS)** in which multiple autonomous agents cooperate to perform household tasks in a smart home environment.

Students will implement, in **Java**, the core components of a MAS:

- a shared **environment**,
- multiple **agents** with different architectures,
- an **organization** and coordination mechanisms,
- interaction mechanisms between agents and between agents and the environment.

This practical work aims to bridge theoretical concepts seen in class (agent architectures, cooperation, organization) with a concrete software implementation.

2 Scenario Description

You are asked to model a **smart apartment** in which autonomous agents collaborate to accomplish daily household tasks.

Typical tasks include (non-exhaustive list):

- cleaning rooms,
- washing dishes,
- throwing out garbage,
- organizing objects,
- monitoring task completion.

Tasks may have constraints such as:

- limited resources (time, energy, tools),
- task dependencies (e.g., garbage must be collected before being thrown out),
- spatial constraints (rooms, locations),
- cooperation requirements (tasks requiring multiple agents).

3 System Requirements

3.1 Environment

The environment represents the apartment and must include:

- a representation of rooms and locations,
- objects (dishes, garbage, cleaning tools, etc.),
- task states (pending, ongoing, completed),
- mechanisms allowing agents to perceive and act on the environment.

The environment may be centralized and shared by all agents.

3.2 Agents

You must implement **at least two different types of agents**, for example:

- **Reactive agents**: agents that respond directly to environmental stimuli using condition–action rules.
- **BDI agents**: agents endowed with *Beliefs*, *Desires*, and *Intentions*, capable of deliberation and planning.

Each agent must:

- perceive the environment,
- decide which task(s) to perform,
- execute actions affecting the environment,
- cooperate or coordinate with other agents.

3.3 Organization and Coordination

Your MAS must include an explicit or implicit form of organization, such as:

- role-based organization (e.g., cleaner, coordinator, assistant),
- task allocation mechanisms,
- cooperation or negotiation protocols.

Coordination strategies may include:

- centralized task assignment,
- distributed decision-making,
- communication-based coordination.

3.4 Agent Interaction

Agents must be able to:

- communicate with each other (direct messages, shared data structures, or events),
- coordinate actions to avoid conflicts and redundancy,
- adapt to changes in the environment (new tasks, completed tasks, failures).

The communication model does not need to follow a specific standard.

4 Implementation Constraints

- Programming language: **Java**
- You are free to use standard Java libraries.
- External MAS frameworks are not allowed.
- The code must be modular, readable, and well-documented. Use Design Patterns !

5 UML Modeling (During the Course)

You must produce a **UML class diagram** representing:

- the environment,
- the different agent classes,
- agent architectures (e.g., BDI components),
- interaction and communication mechanisms.

The UML diagram must reflect the actual design of your Java implementation.

6 Expected Deliverables

6.1 Source Code

- Complete Java source code of the MAS.
- Clear separation between environment, agents, and organizational components.
- A short `README` explaining how to compile and run the project.

6.2 Final Report (Homework)

At the end of this course you will have to submit a written report (**8–12 pages**) including:

- an overview of the system architecture,
- a description of the environment model,
- a detailed presentation of agent types and behaviors,
- the coordination and cooperation mechanisms,
- the UML class diagram,
- a discussion of design choices and limitations.

7 Evaluation Criteria

The project will be evaluated according to the following criteria:

- Correctness and completeness of the MAS implementation,
- Quality of agent cooperation and coordination,
- Appropriateness of agent architectures,
- Clarity and consistency of the UML diagram,
- Code quality and documentation,
- Quality of the final report.

8 Optional Extensions

In the planning course (next course) you will have to extend this work by integrating explictly :

- planning (e.g. Markov Decisions Process)
- learning or adaptation mechanisms (e.g. Reinforcement learning mecasnism),
- formal reasoning components (e.g. logic-based decision-making).