	Document:	MAS practical work
	Course:	ETSE-URV, 2024-25

Table of contents

Practice description

The main goal of the practical work of this course is to put in practice the basic concepts of agent technology in teams of 4-5 students. We will use CrewAI[1], a framework for orchestrating role-playing autonomous AI agents.

The system aims to simulate a city's emergency response problem. The scenario includes various types of agents, which will have to cooperate and coordinate to find the most optimal way to respond to a fire in a city. The ultimate goal is to generate an optimal plan to put the fire out and transport injured people to the nearest hospital.

A fire can start at any time and in any part of the city, posing a significant threat to public safety. The fire can be of various types (ordinary, electrical, gas, etc.) and may require specialized equipment to extinguish.

The city's emergency response system must be designed to efficiently coordinate the response efforts of fire personnel and medical services to put out the fire and treat any injured individuals. The system should take into account the capabilities and locations of the emergency services units, as well as the type and severity of the fire.

Environment

The environment will be a city of your choice that will be represented as a graph. In Lab 4 – Creating custom tools we use the OSMnx[2] package to download and use map data from OpenStreetMap. Moreover, we create a tool that your agents will be able to use to compute the distances between two locations in the city given their coordinates. You will need to use that tool (or an improved version of it) in your practical work.

Inputs

The fires in the map will be provided as inputs to the system. The characteristics of the fire that must be provided are:

- **Fire type**: ordinary, electrical, gas, etc.
- **X Location**: X coordinate where the fire has originated.
- **Y location**: Y coordinate where the fire has originated.
- **Injured people**: how many people has been injured due to the fire.
- **Fire severity**: the severity of the fire, which can be low, medium or high.

Additionally to those inputs, you can include extra information about the fire so that your agents can use those additional details to create a better plan. This data must be provided in a report, that is, the agents must extract the information from a written text in Markdown format.

Note: You must choose a different city as the practical work map. You can use OpenStreetMap to obtain the coordinates of your city of choice.

[1] https://github.com/crewAIInc/crewAI
[2] https://github.com/gboeing/osmnx

The information provided about ambulances, fire trucks and hospitals must be:

- **Id**: the string identifying the ambulance / fire truck / hospital.
- **X Coordinate**: X coordinate of the ambulance / fire truck / hospital.
- **Y Coordinate**: Y coordinate of the ambulance / fire truck / hospital.

Additionally, you must provide additional information. For instance, for fire trucks:

- **Type**: type of fire truck, rescue, ladder, etc.
- **Capacity**: the number of fire people the truck can transport.

For hospitals, you could also provide the number of **available beds**. Those are just examples, you can choose which the additional information will be.

This information must be encoded in a structured file, such as a JSON file.

Agent crews

Several crews of agents are needed to solve this problem. The following crews must be included in your system:

- **Emergency service**: this crew is the one which receives the call asking for help. It has to notify the fire people and medical services about the incidence.
- **Fire people**: this crew is in charge of the fire people units that are distributed throughout the city. It has to choose the most optimal unit or units that will go to the incident area according to the fire type, severity and their capacities to deal with that type of fire.
- **Medical services**: this crew coordinates the medical centres and ambulances of the city. It must decide which is the most optimal health centre to attend the injured people according to the amount of injured people and the distance to that health centre. It must also find the best choice of ambulances.

The final multi-agent system should output a plan which contains all the information to solve this problem. That is, which fire people units and ambulances will respond to the incident and the health centre that will attend the injured people. Additionally, it can include more detailed information, such as the distances, etc.

Requirements

The multi-agent system is open to include different design decisions. However, the system must fulfil a list of minimum requirements to accept the practical work:

1. **Agent crews**: The specified agent crews (emergency service, fire people, and medical services) must be included in the MAS. You are free to add additional crews on top of them.
2. **Flows and routers usage**: Flows and routers must be utilized within the system. You can choose a use case that best suits their application.
3. **Input files creation**: The input files must contain the information specified in the inputs section of this document. The input files you must create are:
    a. City graphml map

   b. Emergency report
   c. Ambulances
   d. Fire trucks
   e. Hospitals
4. **README**: A README.md file must be provided summarising how your MAS works, provided files, custom tools, etc. This file should serve as a reference to understand and use your multi-agent system.
5. **Git**: It is mandatory to document all changes on the practical work using a Git server. You can use GitHub (https://github.com/github), GitLab (https://about.gitlab.com/) or Bitbucket (https://bitbucket.org/). We must have access to the repository to check the evolution of the project. Use the following email address to add the lecturer to the Git repository: jordi.pascual@urv.cat.

Tasks

The first step of the work is the design of the multi-agent system. The design is submitted in Task 1 as detailed below, and the lecturer will provide you feedback about it to know the good and weak points. Then you will have to think about the best coordination mechanisms (Task 2). Finally, you'll have to implement the final system (Final documentation and implementation).

1. Task 1 – Design activity

Description: In this activity, you will perform a first analysis of the practical exercise by defining the characteristics of the multi-agent system. The process involves three steps:

1. **Environment Analysis**: You must analyse and explain the environment in which your agents will operate, according to the properties studied in Lecture 2.
2. **Agent Selection and Definition**: First, you have to decide if you will add additional crews. Then, you will have to decide which agents will form the crews. You can draw inspiration from real-world emergency teams, or choose a different approach based on your own criteria. For each agent, provide a brief description of who it is, its main task, and the tools it has at its disposal. For this task, there's no need to writing an extensive explanation of goals and backstories as we did on the labs.
3. **Agent taxonomy**: Explain the types and properties of all the agents in the multi-agent system, according to the studied taxonomy in Lecture 3.

Delivery content: PDF report with the analysis of the practical exercise; PDF with 8-10 slides for a presentation of up to 10 minutes. No code is required, just a report explaining your design decisions.

Delivery deadline: 3/11/2024

Grade weight: 10%

2. Task 2 - Coordination design

Description: In this task you will define the cooperation and coordination mechanisms that will be used to solve the emergency response problem. There are three main steps:

1. **Process definition**: For each individual crew, define which type of process they will follow (e.g., sequential or hierarchical). You must specify how the sequence of tasks will be executed (or how you envision it to be in the hierarchical case). Moreover, define if some of them are executed in parallel or asynchronously and any other Tasks properties if required (human input, context, etc).
2. **Pydantic Outputs**: Define Pydantic outputs (As seen on Lab 5) for tasks that you think would benefit from structured outputs.
3. **Agent Interaction**: Define how the different agent crews will interact. You must use flows and at least one router (As seen on Lab 8).

Write a detailed explanation of your choices and justification for the decisions you take.

Delivery content: PDF report with the proposed cooperation and coordination mechanisms for the MAS. PDF with 8-10 slides for a presentation of up to 10 minutes.

Delivery deadline: 8/12/2024

Grade weight: 10%

3. Final documentation and implementation

Description: In this activity, you must complete the implementation of the practical work. You must complete the code for the project to execute and provide a solution for the emergency services problem. That includes:

- Writing a README file with brief instructions on how to execute the project, as well as an overview of how the project is configured
- The needed input files

You also need to write a final report with the final design of your system, explaining and justifying all your decisions. You can build it upon the previous Task 1 and 2 reports. If you have made any change over the previous reports, you must specify it and justify it. The report must include some tests on your MAS.

A presentation of your design must also be created. During the presentation, an overview of the code of your project must be shown.

Delivery content: a PDF report with the details of the multi-agent system design and development, including the obtained results; PDF with 10-12 slides for a presentation of up to 12 minutes and the source code of the project in a ZIP file.

Delivery deadline: 12/01/2025

Grade weight: 35%

DISCLAIMER: The final goal of this practical work is to learn the design process of a Multi-Agent System, not to have the best possible output, as this highly depends on the quality of the used LLM. It is more important having a good design with a good justification of your choices than an excellent emergency plan. The results will also be part of the evaluation, but will not be the only evaluation criteria.

4. Teamwork

Team work will be evaluated too, as one of the advantages of multi-agent systems is to facilitate working in a distributed way. The team must write minutes of the meetings of the group using the forum task available in the URV Virtual Campus. The minutes must indicate how the work is planned and distributed into the different team members, re-planning actions, changes introduced during the project and any other important issue and decisions related with the group work. They are open to include other additional information, such as members of the team present in the meeting, date and location of the meeting, etc. The minutes forum is only accessible by the group and the lecturers.

You can assign different roles in the team if you want: person in charge of the minutes (i.e., secretary), person in charge of reporting, leader, etc.

If someone abandons the course, please write it in the minutes, but also email the lecturers as soon as possible.

Grade weight: 5%

Useful links and recommended readings

It is useful to use well-known and widely used tools such as:

- As IDE, VS Code (https://code.visualstudio.com/)
- CrewAI (https://docs.crewai.com/)
- CrewAI examples (https://github.com/crewAIInc/crewAI-examples)
- Ollama (https://ollama.com/)