Universitat Rovira i Virgili

Escola Tècnica Superior d'Enginyeria

Task 2 - Coordination design

Design of a MAS for emergency Response Simulation in Urban Environments

Autors:

Pau Baguer Fàbrega

Niklas Long Schiefelbein

Kacper Poniatowski

Carlos Jiménez Farfán

Pedro Agundez Fernández

December 2024

Contents

1 Introduction

This report presents the work undertaken for Task 2, which builds upon the foundational design of the Multi-Agent System (MAS) developed in Task 1. The MAS is intended to manage emergency responses in an urban environment by coordinating a network of autonomous agents. The primary objective of this project is to design an efficient, collaborative framework for handling rescue scenarios, including fire outbreaks, medical emergencies, and traffic disruptions. The selected environment emulates real-world complexities, emphasizing resource optimization to ensure timely and effective responses by the emergency crews.

Task 2 focuses on advancing the design by addressing the following key aspects:

1) Defining the specific tasks assigned to each crew and the distribution of responsibilities among the agents within each crew.

2) Establishing collaborative workflows and negotiation mechanisms for agents operating within the same crew.

3) Designing the data structures for task outputs using Pydantic models to ensure consistency and validation.

4) Developing high-level information exchange flows between different crews to facilitate inter-crew collaboration.

This analysis seeks to establish a cohesive relationship between agents and their respective crews, which is essential for achieving the overarching objectives of the MAS. Each agent is equipped with specialized capabilities, enabling the system to formulate tailored solutions to the diverse and dynamic challenges posed by emergency scenarios.

1.1 MAS Changes from Task 1

Our MAS has undergone updates to better align with the requirements of this and future tasks. Specifically, agents have been further specialized, enhancing their expertise in specific domains and equipping them with state-specific knowledge of the environment, such as real-time hospital occupancy or the availability of fire trucks.

The following subsections outline these changes in greater detail.

1.1.1 Firefighting Crew

The firefighting crew now comprises six specialized agents, each with a distinct role. The crew operates within a flat organizational structure, meaning that no single agent acts as a superior issuing direct orders. Instead, all agents collaboratively contribute to achieving the crew's objectives.

Additionally, lower-ranked firefighters are included in the system as supplementary resources. While these individuals actively participate in the execution of tasks described in the plan, they do not contribute to the planning process itself and are therefore not formally integrated into the firefighting crew.

The following agents have been newly designed to address the complex demands of emergency scenarios. Each agent possesses specialized expertise to ensure efficient and effective decision-making during crisis situations:

1. **Call Receiver and Personnel Specialist**: Receives the emergency call from the Emergency crew. Responsible for determining the number of task forces required to manage the emergency effectively.
2. **Fire Truck Chief**: Maintains real-time knowledge of fire truck locations and selects the nearest trucks to the incident. Additionally, the chief identifies any supplementary trucks required and designs optimal navigation routes for deployment.
3. **Tool Specialist**: Decides on the necessary equipment based on the fire type and other variables. This includes selecting tools such as ladders, specialized protective suits for extreme conditions, and ventilation equipment.
4. **Fire Type Specialist Chief**: Selects appropriate extinguishing tools tailored to the fire type. For instance, CO_2 extinguishers are chosen for electrical fires, while other methods are applied for different scenarios.
5. **Rescue Specialist**: Evaluates the condition of victims and prioritizes rescue operations based on the severity of their situations.
6. **Building Structure Specialist**: Assesses the structural integrity of the building and provides recommendations for victim rescue. This may involve deciding whether to enter the building or deploy external equipment such as a large set of stairs.

1.1.2 Medical Crew

The medical crew is composed of five specialized agents, each with a distinct role aimed at ensuring an effective and efficient response to medical emergencies. Like the firefighting crew, the medical team operates within a flat organizational structure, fostering collaborative decision-making without hierarchical directives.

Additionally, lower-ranked medical personnel, such as paramedics and support staff, are included in the system as supplementary resources. These individuals actively assist in executing tasks outlined in the plan but do not contribute to the planning process and are therefore not formally integrated into the medical crew.

The following agents have been newly designed to meet the unique challenges posed by medical emergencies. Each agent brings specialized knowledge and skills to the operation:

1. **Call Receiver and Personnel Specialist**: Receives the emergency call from the Emergency crew. Responsible for determining the number of task forces required to manage the medical emergency effectively.
2. **Triage Specialist**: Assesses the condition of victims at the scene and assigns treatment priorities based on the severity of their injuries, ensuring that the most critical cases are addressed first.
3. **Ambulance Coordinator**: Maintains real-time knowledge of ambulance availability and locations, assigning the closest and most suitable ambulances to transport

patients to medical facilities efficiently. He also computes the fastest route towards the emergency site.

4. **Hospital Coordinator**: Monitors the status of hospitals, including bed availability and specialization capabilities, to ensure victims are directed to the most appropriate facility for their needs.
5. **Equipment Specialist**: Determines the necessary medical supplies and equipment required at the scene, such as stretchers, defibrillators, or oxygen tanks, ensuring they are deployed promptly.

1.1.3 Police Crew

The police crew consists of three specialized agents, each playing a vital role in managing the response to emergencies. This crew operates within a flat organizational structure, allowing all agents to contribute collaboratively without reliance on hierarchical directives.

In addition to these agents, lower-ranked personnel, such as field officers and auxiliary staff, are included as supplementary resources. While these individuals assist in the execution of tasks, they do not participate in the planning process and are therefore not formally integrated into the police crew.

The following agents have been designed to address the unique challenges of law enforcement and traffic management during emergency scenarios:

1. **Receiver and Personnel Specialist**: Evaluates the nature of the emergency and determines the number of taskforces required to manage the situation effectively.
2. **Dispatch Specialist**: Identifies the patrol cars nearest to the incident and develops optimal navigation plans to ensure timely arrival at the scene.
3. **Traffic Specialist**: Analyzes the streets surrounding the emergency site and devises roadblock strategies to manage traffic, ensuring a safe and secure environment for emergency operations.

1.1.4 Emergency Crew

The emergency crew comprises three specialized agents, each playing a critical role in initiating and managing the response to emergencies. This crew serves as the primary point of coordination for distributing tasks among other specialized crews (firefighting, medical, and police).

Task 1 comprised a fifth crew, integrating the philosopher agent, who would provide intel on the ethics of the planned actions. It has been decided to integrate this agent into the emergency crew, limiting the number of crews of the MAS to four.

As with the other crews, supplementary resources, such as administrative staff or support personnel, may assist with execution but are not formally integrated into the emergency crew's planning process.

The following agents have been designed to handle the unique challenges of emergency management and coordination:

1. **Caller/Receiver Agent**: Acts as the first point of contact, receiving initial reports or queries from the public or emergency systems. This agent processes the information and passes it on to the distributor for further action.
2. **Distributor**: Analyzes the received query, determines the nature of the emergency, and distributes it into specific queries for the relevant crews (firefighting, medical, or police). This agent also decides the appropriate routing for cases requiring medical attention.
3. **Philosopher**: Monitors for ethical dilemmas that may arise in the plans developed by any crew (firefighting, medical, or police). This agent provides recommendations to ensure the plans align with ethical standards and prioritize human safety and fairness.

2 Process definition

This section defines the internal workflows of each crew of agents. Each crew is assigned specific tasks that constitute their individual objectives. Given the completely flat organizational structure—where no agent is designated to assign tasks to others—all crews operate within a sequential framework. Tasks are executed one after another in the order outlined in this report, ensuring dependencies between tasks are addressed effectively. Furthermore, as each agent has been precisely defined, tasks are directly assigned to the appropriate agents in accordance with the sequential order of operations.

2.1 Firefighting Crew

Definition of Tasks to be performed by the Firefighting Crew:

1) **Assign Taskforce: Personnel Specialist** performs the task of determining the number of firefighters required for the emergency. This involves analyzing inputs such as the fire type, fire location, the number of injured people, and the estimated severity. The output includes the number of firefighters needed (`n_firefighters`) and a natural language description (`action_details`) detailing the taskforce allocation.

2) **Select Extinguishing Tools: Fire Type Specialist** performs the task of selecting the appropriate tools for extinguishing the fire. The input is the fire type, and the output includes a list of tools (`extinguishing_tools`) and a natural language explanation (`action_details`) of the chosen tools and their purposes.

3) **Assess Building Structure: Building Structure Specialist** performs the task of evaluating the structural integrity of the building at the fire location. The input is the fire location, and the outputs include a rescue strategy (`rescue_strategy`), a fire suppression strategy (`stop_fire_strategy`), and a natural language description (`action_details`) detailing these strategies.

4) **Plan Victim Rescue: Rescue Specialist** performs the task of prioritizing the rescue of injured individuals. The input is the number and condition of injured people, and the output includes a prioritized list of victims for rescue (`ordered_victim_rescue`) and a detailed explanation (`action_details`) of the prioritization process.

5) **Select Tools: Tool Specialist** performs the task of selecting the equipment needed for rescue and fire suppression. The inputs include the rescue strategy and fire suppression strategy. The output consists of a list of firefighting tools (`firefighting_tools`) and a natural language explanation (`action_details`) of their selection and use.

6) **Select Fire Trucks: Fire Truck Chief** performs the task of identifying and assigning fire trucks for deployment. The inputs are the fire location and the required extinguishing tools. The outputs include the selected fire trucks (`fire_trucks`) and a description (`action_details`) of the selection process.

7) **Plan Route: Fire Truck Chief** performs the task of determining the most efficient routes for fire trucks to reach the emergency site. The inputs include the

fire trucks assigned and the fire location. The output consists of optimized routes (`routes`) and a natural language explanation (`action_details`) of the routing plan.

8) **Compile Plan: Personnel Specialist** performs the task of consolidating all information into a comprehensive firefighting response plan. The inputs include all outputs from the previous tasks, such as the number of firefighters, extinguishing tools, rescue strategy, fire suppression strategy, victim rescue order, tools, fire trucks, routes, and any ethical issues identified during the process. The output is a detailed response plan (`response_plan`) that incorporates all these elements to guide the firefighting operation.(`action_details`) contains the concatenated details of each task. An additional output (`ethical_issues`) lists possible ethical issues that the firefighters may have during the planning process. Theses will be checked by the ethical agent from the emergency crew and will add recommendations on them.

Task	Agent	Input	Output
Assign taskforce	Personnel Specialist	*fire_type* *fire_location* *injured_people* *estimated_serverity*	*n_firefighters* *action_details*
Select extinguishing tools	Fire Type Specialist	*fire_type*	*extinguishing_tools* *action_details*
Assess building structure	Building Structure Specialist	*fire_location*	*rescue_strategy* *stop_fire_strategy* *action_details*
Plan victim rescue	Rescue Specialist	*injured_people*	*ordered_victim_rescue* *action_details*
Select tools	Tool Specialist	*rescue_strategy* *stop_fire_strategy*	*firefighting_tools* *action_details*
Select fire trucks	Fire Truck Chief	*fire_location* *extinguishing_tools*	*fire_trucks* *action_details*
Plan route	Fire Truck Chief	*fire_trucks* *fire_location*	*routes* *action_details*
Compile plan	Personnel Specialist	All aforementioned outputs.	*response_plan* *ethical_issues* *action_details*

Table 1: Inputs and outputs of tasks performed by the Firefighting crew.

2.2 Medical Crew

Definition of Tasks to be performed by the Medical Crew:

1) **Assign Taskforce: Personnel Specialist** performs the task of determining the number of doctors required for the emergency. This involves analyzing inputs such as the number of injured people, the fire location, the fire type, and the estimated severity of the incident. The output includes the number of doctors needed (`n_doctors`) and a natural language description (`action_details`) detailing the taskforce allocation.

2) **Compile Medical Supplies: Equipment Specialist** performs the task of selecting and preparing the necessary medical supplies for the emergency. The input is

the fire type, which informs the choice of supplies. The output includes the compiled list of medical supplies (`medical_supplies`) and a description (`action_details`) of the chosen items and their relevance to the emergency.

3) **Check Hospital Capacity: Hospital Coordinator** performs the task of evaluating the availability of hospital resources to accommodate the injured. **The availability is given via Human input**. The input is the number of injured people, and the output includes the capacities of nearby hospitals (`hospital_capacities`) and a description (`action_details`) of the hospital capacity assessment.

4) **Hospital Voting: Hospital Coordinator** performs the task of coordinating a voting mechanism among hospitals to determine which facilities can accept the victims. This tasks starts a variation on a Borda voting procedure between the Hospital Coordinator and the Triage Specialist. The principle behind this voting is that the Hospital Coordinator will prioritize hospitals with less capacity, but the Triage Specialist will prioritize sending the victims with more severe wounds to the best hospitals with more specialists and equipment. Therefore, to begin with, the Hospital Coordinator will rank Hospitals based on their capacity following $votes = [N_{hospials}, N_{hospials} - 1, \ldots, 0]$. The input is the hospital capacities, and the output includes the voting results (`hospital_votes`) and a natural language explanation (`action_details`) of the voting process.

5) **Victim Voting: Triage Specialist** performs the task of mapping victims to hospitals based on the severity of their injuries and hospital availability. This tasks follows up on the Borda voting descrived in the previous task. Now, the Triage Specialist will vote for each victim, based on the severity of its wounds, he will place $N_{hospials}$ votes with preferrence to send a particular victim to a hospital or another. Therefore, a victim's vote is as follows: $votes_{victim_i} = [N_{hospials}, N_{hospials} - 1, \ldots, 0]$, and all votes are $votes = [votes_{victim_1}, votes_{victim_2}, \ldots, votes_{victim_{N_{victims}}}]$. The input is the number of injured people, and the output includes a mapping of victims to hospitals (`victim_hospital_voting_map`) and an explanation (`action_details`) of the prioritization strategy.

6) **Select Ambulances: Ambulance Coordinator** performs the task of identifying and assigning ambulances for transporting victims to hospitals. The inputs are the fire location and the required medical supplies. The output includes the list of selected ambulances (`ambulances`) and a description (`action_details`) of the selection process.

7) **Plan Route: Ambulance Coordinator** performs the task of determining the most efficient routes for ambulances to transport victims to hospitals. The inputs are the selected ambulances and the fire location. The output includes the planned routes (`routes`) and a natural language description (`action_details`) of the routing plan.

8) **Compile Plan: Personnel Specialist** performs the task of consolidating all information into a comprehensive medical response plan. The inputs include all outputs from the previous tasks, such as the number of doctors, medical supplies, ambulances, routes, and victim-hospital mapping. The latter variable, is the result of the Borda voting process. In this task, the Personnel Specialist is going to determine

the winning hosptial for each victim, summing up the votes from the Hospital Coordinator and the Triage Specialist. Afterwards, he is also going to make sure that after this assignation, there is still bed capacity in all hospitals. The output is a detailed response plan (`response_plan`) that incorporates all these elements to guide the medical operation. (`action_details`) contains all the compiled details for the different tasks. An additional output (`ethical_issues`) lists possible ethical issues that the firefighters may have during the planning process. Theses will be checked by the ethical agent from the emergency crew and will add recommendations on them.

Task	Agent	Input	Output
Assign taskforce	Personnel Specialist	*injured_people* *fire_location* *fire_type* *estimated_severity*	*n_doctors* *action_details*
Compile medical supplies	Equipment Specialist	*fire_type*	*medical_supplies* *action_details*
Check hospital capacity	Hospital Coordinator	*injured_people*	*hospital_capacities* *action_details*
Hospital Voting	Hospital Coordinator	*hospital_capacities*	*hosptial_votes* *action_details*
Victim Voting	Triage Specialist	*injured_people*	*victim_hospital_voting_map* *action_details*
Select Ambulances	Ambulance Coordinator	*fire_location* *medical_supplies*	*ambulances* *action_details*
Plan route	Ambulance Coordinator	*fire_trucks* *fire_location*	*routes* *action_details*
Compile plan	Personnel Specialist	All aforementioned outputs.	*response_plan* *ethical_issues* *action_details*

Table 2: Inputs and outputs of tasks performed by the Medical crew.

2.3 Police Crew

1) **Assign Taskforce: Receiver and Personnel Specialist** evaluates the nature of the emergency and determines the number of police officers required. The inputs include the fire location and estimated severity. The outputs include the number of officers needed (`n_officers`) and a natural language description (`action_details`) of the evaluation.

2) **Plan Perimeter Control: Traffic Specialist** analyzes the streets around the emergency site and plans safety perimeters to secure the area. The inputs include the fire location. The outputs are the proposed perimeter locations (`perimeter_locations`) and an explanation (`action_details`) of the perimeter strategy.

3) **Assign Vehicles: Dispatch Specialist** identifies patrol cars nearest to the incident and assigns them to tasks such as perimeter monitoring and evacuation support. The inputs are the fire location and perimeter locations. The outputs include the assigned vehicles (`patrol_vehicles`) and a description (`action_details`) of their allocation.

4) **Plan Routes: Dispatch Specialist** determines the most efficient routes for patrol vehicles to monitor safety perimeters or assist in evacuations. The inputs are the assigned vehicles and perimeter locations. The outputs include optimized patrol routes (`patrol_routes`) and an explanation (`action_details`) of the routing logic.

5) **Compile Police Plan: Receiver and Personnel Specialist** consolidates all information into a comprehensive police response plan. The inputs include all outputs from previous tasks, such as perimeter locations, assigned vehicles, routes, and coordination summary. The outputs are a detailed response plan (`action_details`) and any identified ethical issues (`ethical_issues`).

Task	Agent	Input	Output
Assess Situation	Personnel Specialist	*fire_location* *estimated_severity*	*n_officers* *action_details*
Plan Perimeter Control	Traffic Specialist	*fire_location*	*perimeter_locations* *action_details*
Assign Vehicles	Dispatch Specialist	*fire_location* *perimeter_locations*	*patrol_vehicles* *action_details*
Plan Routes	Dispatch Specialist	*patrol_vehicles* *perimeter_locations*	*patrol_routes* *action_details*
Compile Police Plan	Personnel Specialist	*All aforementioned outputs*	*response_plan* *ethical_issues* *action_details*

Table 3: Inputs and outputs of tasks performed by the Police crew.

2.4 Emergency Crew

1) **Asses and Distribute Emergency Alert: Distributor** acts as the first point of contact, receiving the initial emergency alert. The Distributor assesses the scale of the emergency based on the received details and potential risks and assigns specific tasks to relevant crews (firefighting, medical, police) based on the emergency type. The input is the markdown file, including fire type and all initial information. The outputs include distributed tasks (`distributed_tasks`) and a detailed explanation (`action_details`) of the distribution logic.

2) **Compile Situation Report: Distributor** consolidates all subplans from each crew into a comprehensive situation report. The inputs include plans from the other crews. The outputs are a detailed situation report (`situation_report`) and any remaining ethical concerns (`ethical_issues`).

3) **Ethical Consultation and Final Compilation: Philosopher** evaluates the consolidated plans developed by other crews to resolve any ethical concerns. The inputs include the detailed situation report and any remaining ethical concerns. The output consists of the detailed situation report including suggestions to resolve the ethical concerns (`situation_report`).

Task	Agent	Input	Output
Assess and Distribute Emergency Alert	Distributor	*markdown file*	*distributed_tasks* *action_details*
Compile Situation Report	Distributor	*plans from other crews*	*situation_report* *ethical_issues*
Ethical Consultation and final Compilation	Philosopher	*detailed situation report* *remaining ethical concerns*	*situation_report*

Table 4: Inputs and outputs of tasks performed by the Emergency crew.

3 Pydantic Outputs

3.1 Firefighting Crew

1) Assign Taskforce:

```python
class TaskforceAssignment(BaseModel):
    """Output for assigning the taskforce."""

    n_firefighters: int = Field(..., description="Number of firefighters assigned.")
    action_details: str = Field(
        ..., description="Additional details of the assignment action."
    )
```

Listing 1: TaskforceAssignment Firefighting Pydantic Output

2) Select Extinguishing Tools:

```python
class ExtinguishingToolSelection(BaseModel):
    """Output for selecting extinguishing tools."""

    extinguishing_tools: List[str] = Field(
        ..., description="List of selected tools for extinguishing."
    )
    action_details: str = Field(
        ..., description="Details of the tool selection process."
    )
```

Listing 2: ExtinguishingToolSelection Pydantic Output

3) Assess Building Structure:

```python
class AssessBuildingStructurePlanning(BaseModel):
    """Output for assessing the building structure."""

    rescue_strategy: str = Field(..., description="Strategy for rescuing victims.")
    stop_fire_strategy: str = Field(..., description="Strategy for stopping the fire.")
    action_details: str = Field(
        ..., description="Further action details for building assessment."
    )
```

Listing 3: AssessBuildingStructurePlanning Pydantic Output

4) Plan Victim Rescue:

```python
class VictimRescuePlanning(BaseModel):
    """Output for planning victim rescue."""

    ordered_victim_rescue: List[str] = Field(
        ..., description="List of victims ordered by rescue priority."
    )
    action_details: str = Field(
        ..., description="Details of the rescue planning process."
    )
```

Listing 4: VictimRescuePlanning Pydantic Output

5) Select Tools:

```python
class ToolSelection(BaseModel):
    """Output for selecting firefighting tools."""

    firefighting_tools: List[str] = Field(
        ..., description="List of selected firefighting tools."
    )
    action_details: str = Field(
        ..., description="Details about the tool selection process."
    )
```

Listing 5: ToolSelection Pydantic Output

6) Select Fire Trucks:

```python
class FireTruckSelection(BaseModel):
    """Output for selecting fire trucks."""

    fire_trucks: List[str] = Field(..., description="List of selected fire
        trucks.")
    action_details: str = Field(
        ..., description="Details of the fire truck selection process."
    )
```

Listing 6: FireTruckSelection Pydantic Output

7) Plan Route:

```python
class RoutePlanning(BaseModel):
    """Output for planning the route."""

    routes: List[tuple[str, List[int]]] = Field(
        ..., description="List of planned OSMnx routes for each fire truck."
    )
    action_details: str = Field(
        ..., description="Additional information on route planning."
    )
```

Listing 7: RoutePlanning Firefighting Pydantic Output

8) Compile Plan:

```python
class FirePlanCompilation(BaseModel):
    """Comprehensive output for compiling the final plan."""

    response_plan: str = Field(
        ..., description="Compiled response plan for the firefighter operation."
    )
    ethical_issues: Optional[str] = Field(
        None, description="Ethical issues considered in the plan."
    )
    action_details: str = Field(
        ..., description="Additional details of the overall firefighter plan."
    )
```

Listing 8: FirePlanCompilation Pydantic Output

3.2 Medical Crew

1) Assign Taskforce:

```python
class TaskforceAssignment(BaseModel):
    """Output for assigning the taskforce."""

    n_doctors: int = Field(..., description="Number of doctors assigned.")
    action_details: str = Field(
        ..., description="Additional details of the assignment action."
    )
```

Listing 9: TaskforceAssignment Medical Pydantic Output

2) Compile Medical Supplies:

```python
class MedicalSupplySelection(BaseModel):
    """Output for compiling medical supplies."""

    medical_supplies: List[str] = Field(
        ..., description="List of compiled medical supplies."
    )
    action_details: str = Field(
        ..., description="Details of the medical supplies compilation process."
    )
```

Listing 10: MedicalSupplySelection Pydantic Output

3) Check Hospital Capacity:

```python
class HospitalCapacityCheck(BaseModel):
    """Output for checking hospital capacity."""

    hospital_capacities: List[int] = Field(
        ..., description="List of hospital capacities."
    )
    action_details: str = Field(
        ..., description="Details of the hospital capacity check."
    )
```

Listing 11: HospitalCapacityCheck Pydantic Output

4) Hospital Voting:

```python
class HospitalVoting(BaseModel):
    """Output for hospital voting."""

    hospital_votes: List[int] = Field(..., description="Votes for each hospital.")
    action_details: str = Field(
        ..., description="Details of the hospital voting process."
    )
```

Listing 12: HospitalVoting Pydantic Output

5) Victim Voting:

```python
class VictimVoting(BaseModel):
    """Output for victim voting."""

    victim_hospital_voting_map: dict = Field(
        ..., description="Mapping of victims to hospital votes."
    )
    action_details: str = Field(
        ..., description="Details of the victim voting process."
    )
```

Listing 13: VictimVoting Pydantic Output

6) Select Ambulances:

```python
class AmbulanceSelection(BaseModel):
    """Output for selecting ambulances."""

    ambulances: List[str] = Field(..., description="List of selected ambulances.")
    action_details: str = Field(
        ..., description="Details of the ambulance selection process."
    )
```

Listing 14: AmbulanceSelection Pydantic Output

7) Plan Route:

```python
class RoutePlanning(BaseModel):
    """Output for planning the route."""

    routes: List[tuple[str, List[int]]] = Field(
        ..., description="List of planned OSMnx routes for each ambulance."
    )
    action_details: str = Field(
        ..., description="Details of the route planning process."
    )
```

Listing 15: RoutePlanning Medical Pydantic Output

8) Compile Plan:

```python
class MedicalPlanCompilation(BaseModel):
    """Comprehensive output for compiling the final plan."""

    response_plan: str = Field(
        ..., description="Compiled response plan for the medical operation."
    )
    ethical_issues: Optional[str] = Field(
        None, description="Ethical issues considered in the plan."
    )
    action_details: str = Field(
        ..., description="Additional details of the overall medical plan."
    )
```

Listing 16: MedicalPlanCompilation Pydantic Output

3.3 Police Crew

1) Assign Taskforce:

```python
class TaskforceAssignment(BaseModel):
    """Output for assigning the taskforce."""

    n_officers: int = Field(
        ..., description="Number of officers assigned to the situation."
    )
    action_details: str = Field(
        ..., description="Details of the situation assessment action."
    )
```

Listing 17: TaskforceAssignment Police Pydantic Output

2) Plan Perimeter Control:

```python
class PerimeterControlPlanning(BaseModel):
    """Output for planning perimeter control."""

    perimeter_locations: List[str] = Field(
        ..., description="List of locations defining the perimeter."
    )
    action_details: str = Field(
        ..., description="Details of the perimeter control plan."
    )
```

Listing 18: PerimeterControlPlanning Pydantic Output

3) Assign Vehicles:

```python
class PatrolSelection(BaseModel):
    """Output for assigning vehicles."""

    patrol_vehicles: List[str] = Field(
        ..., description="List of assigned patrol vehicles."
    )
    action_details: str = Field(
        ..., description="Details of the vehicle assignment process."
    )
```

Listing 19: PatrolSelection Pydantic Output

4) Plan Routes:

```python
class RoutePlanning(BaseModel):
    """Output for planning patrol routes."""

    patrol_routes: List[tuple[str, List[int]]] = Field(
        ..., description="List of planned patrol routes."
    )
    action_details: str = Field(
        ..., description="Details of the route planning process."
    )
```

Listing 20: RoutePlanning Pydantic Output

5) Compile Police Plan:

```python
class PolicePlanCompilation(BaseModel):
    """Comprehensive output for compiling the police plan."""

    response_plan: str = Field(
        ..., description="Compiled response plan for the police operation."
    )
    ethical_issues: Optional[str] = Field(
        None, description="Ethical issues considered in the plan."
    )
    action_details: str = Field(
        ..., description="Additional details of the overall police plan."
    )
```

Listing 21: PolicePlanCompilation Pydantic Output

3.4 Emergency Crew

1) Asses and Distribute Emergency Alert:

```python
class EmergencyAlertDistribution(BaseModel):
    """Output for assessing and distributing emergency alerts."""

    distributed_tasks: List[str] = Field(
        ..., description="List of distributed tasks from the emergency alert."
    )
    action_details: str = Field(
        ..., description="Details of the distribution process for emergency
            alerts."
    )
```

Listing 22: EmergencyAlertDistribution Pydantic Output

2) Compile Situation Report:

```python
class SituationReportCompilation(BaseModel):
    """Output for compiling the situation report."""

    situation_report: str = Field(..., description="Compiled situation report.")
    ethical_issues: Optional[str] = Field(
        None, description="Ethical issues identified in the situation report."
    )
```

Listing 23: SituationReportCompilation Pydantic Output

3) Ethical Consultation and Final Compilation:

```python
class FinalCompilation(BaseModel):
    """Output for ethical consultation and final compilation."""

    situation_report: str = Field(
        ..., description="Finalised situation report after ethical consultation."
    )
```

Listing 24: FinalCompilation Pydantic Output

4 Agent Interaction

This section discusses the interaction of agents and crews within the system. The workflow involves multiple crews interacting with one another to ensure a coordinated and effective resolution plan is generated for the given emergency situation.

The following subsections discuss the workflow in detail, starting from the creation of the *UserRequest* to the generation of the *FinalPlan* output.

4.1 Emergency Crew

This subsection discusses the individual agents that make up the Emergency Crew. This detailed explanation is necessary due to the more complex interactions between the Emergency Crew and other crews, and it provides a deeper understanding of how these agents collaborate within the system.

4.1.1 Distributor Agent

The Distributor Agent serves as the central decision-making and coordination entity within the Emergency Crew. Upon receiving a *UserRequest*, it distributes relevant information to the specialized crews according to the nature of the emergency. The Distributor Agent initiates queries and ensures appropriate crews are engaged.

- **Firefighting Crew Interaction**: The Distributor Agent forwards information about fire-related emergencies to the Firefighting Crew.
- **Police Crew Interaction**: The Distributor Agent relays necessary details to the Police Crew.
- **Medical Crew Interaction**: The Distributor Agent checks whether injured people are involved. If there are injured people, the relevant information is sent to the Medical Crew. If no injuries are reported, the medical planning process is skipped, and the Receiver Agent moves forward with the plans of other crews.

4.1.2 Receiver Agent

The Receiver Agent, part of the Emergency Crew, is responsible for aggregating the plans generated by the Firefighting, Police, and Medical Crews. It combines these individual plans into a comprehensive *FinalPlan* to address the emergency. The Receiver ensures that all plans are considered, including the acknowledgment of skipped steps (e.g., no medical plan).

4.1.3 Philosopher Agent

Before finalizing the response, the Receiver Agent evaluates the compiled *FinalPlan* for potential ethical concerns. If ethical issues are detected, the Philosopher Agent is consulted to provide recommendations which are incorporated into the output. If no ethical issues are identified, the workflow bypasses the Philosopher Agent, and the *FinalPlan* is finalized without additional consultation.

4.2 Medical Crew

The Medical Crew is tasked with addressing situations involving injured individuals. When injuries are present in the *UserRequest* it sends relevant information to the Medical Crew, which then generates a *MedicalPlan*. This plan is subsequently sent to the Receiver Agent for integration. If no injuries are detected, the Distributor bypasses the Medical Crew and acknowledges that no medical plan is required.

4.3 Firefighting Crew

The Firefighting Crew is responsible for handling fire-related details of the emergency response. Upon receiving information from the Distributor Agent, the Firefighting Crew processes the provided details and assesses the situation. Based on the information, it develops a *FirefightingPlan*, which outlines the specific steps required to address the fire emergency effectively. Once the plan is complete, the Firefighting Crew sends the *FirefightingPlan* to the Receiver Agent for integration into the comprehensive *FinalPlan*.

4.4 Police Crew

The Police Crew manages the law enforcement aspects of the emergency. The Police Crew analyses the situation and reviews the details provided by the Distributor Agent. Using this information, the Police Crew creates a *PolicePlan*, which specifies the actions required to address the law enforcement components of the emergency. After completing the plan, the Police Crew sends the *PolicePlan* back to the Receiver Agent to be integrated into the *FinalPlan*.

4.5 Design Decisions

- We represented the Emergency Crew as two separate entities within the Mermaid diagram to maintain readability and ensure the flow remains as clear as possible. However, it is important to note that the Emergency Crew is considered a single unified entity in reality.
- We utilized two individual routers within our workflow, as specified in the assignment brief. The first router determines whether the Medical Crew is required based on the information provided in the *UserRequest*. The second router assesses whether any of the individual crews, during the creation of their respective plans, detected ethical issues that need to be addressed. If ethical concerns are identified, the Philosopher Agent is consulted to resolve these issues. If no ethical concerns are detected, the Philosopher Agent is bypassed.

4.6 Flow Diagram

Figure 1 displays the entire flow of the Multi Agent System:

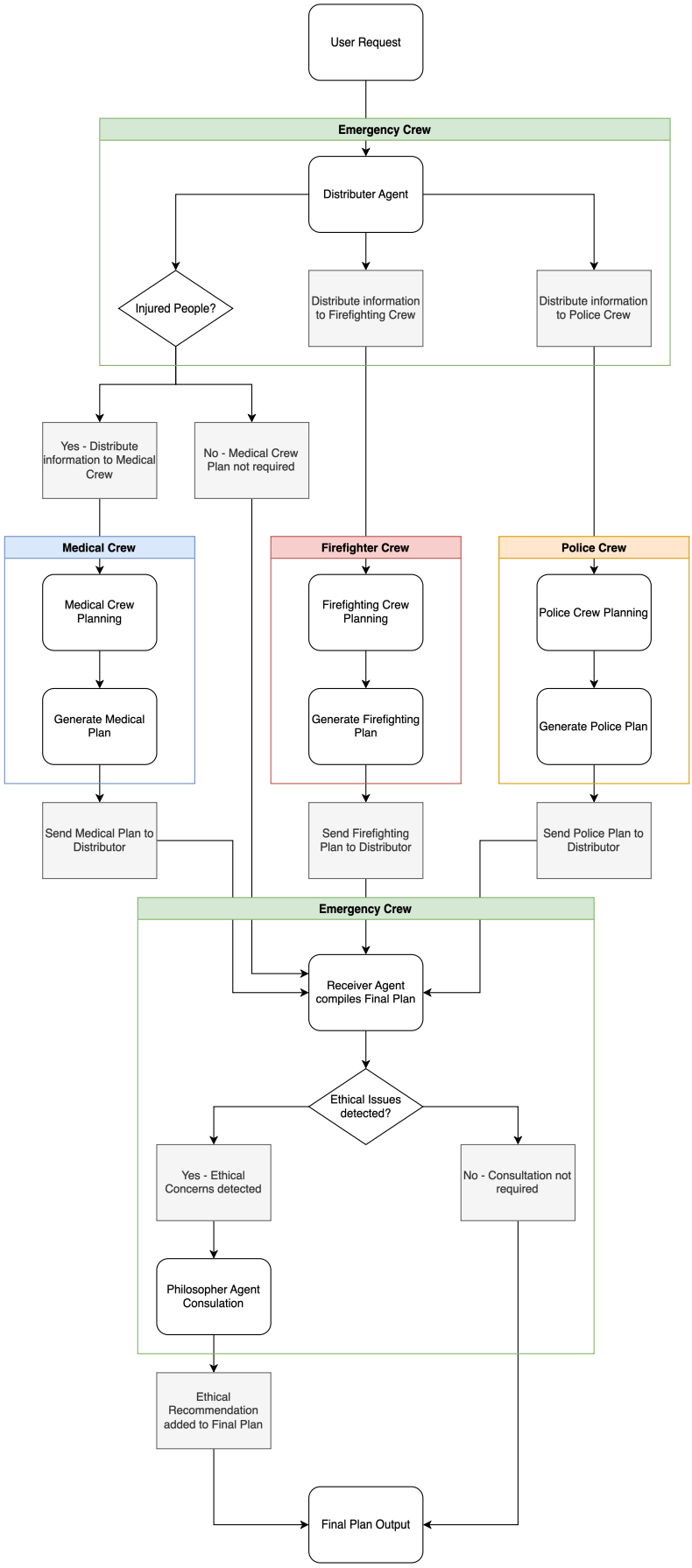

Figure 1: Flow diagram showing the emergency workflow.

5 Conclusions

The updated MAS demonstrates a robust coordination framework across four specialized crews: Firefighting, Medical, Police, and Emergency. Each crew operates independently within a flat hierarchy, ensuring streamlined workflows and specialized task execution. The detailed definition and assignment of agents to specific tasks enhance the system's ability to address complex emergencies. By leveraging agents' domain-specific expertise, the MAS can formulate precise and efficient response plans.

The sequential structure ensures that dependencies between tasks are respected, allowing each task's outputs to seamlessly feed into subsequent operations. This design minimizes delays and reduces the risk of errors in planning. Interaction between crews, facilitated by the Emergency Crew's Distributor and Receiver Agents, ensures comprehensive and coordinated response plans. The inclusion of the Philosopher Agent for ethical considerations highlights the system's adaptability to nuanced challenges.

By integrating the Philosopher Agent to address potential ethical issues, the MAS ensures that response plans are not only operationally effective but also align with ethical standards, fostering trust and fairness in emergency management. The implementation of Pydantic models ensures consistency and validation of task outputs, promoting data integrity and facilitating smooth integration between agents and crews.

The MAS design demonstrates scalability by defining workflows that can adapt to varying emergency scales and complexities. Its flat structure and modular task assignments allow flexibility in managing diverse scenarios. The system's detailed task definitions, inter-crew collaboration mechanisms, and ethical safeguards provide a strong foundation for further enhancements, such as the integration of real-time data analytics and machine learning models to improve decision-making.

References

[1] Parasumanna Gokulan Balaji and Dipti Srinivasan. An introduction to multi-agent systems. *Innovations in multi-agent systems and applications-1*, pages 1–27, 2010.

[2] Yiling Chen. *Voting Protocols*. Hardvard, 2011.

[3] Katae Larson. *Multiagent Systems: Introduction to Social Choice*. U. Waterloo.

[4] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[5] Wenrui Zou. Overview on reinforcement learning of multi-agent game. In *Journal of Physics: Conference Series*, volume 2646, page 012021. IOP Publishing, 2023.