UNIVERSITÀ
DI TRENTO
Dipartimento di
Ingegneria e Scienza dell’Informazione

Autonomous Software Agents
Assignment A2

Marco Robol

marco.robol@unitn.it

Simulating the vaccine distribution scenario in UnityProlog

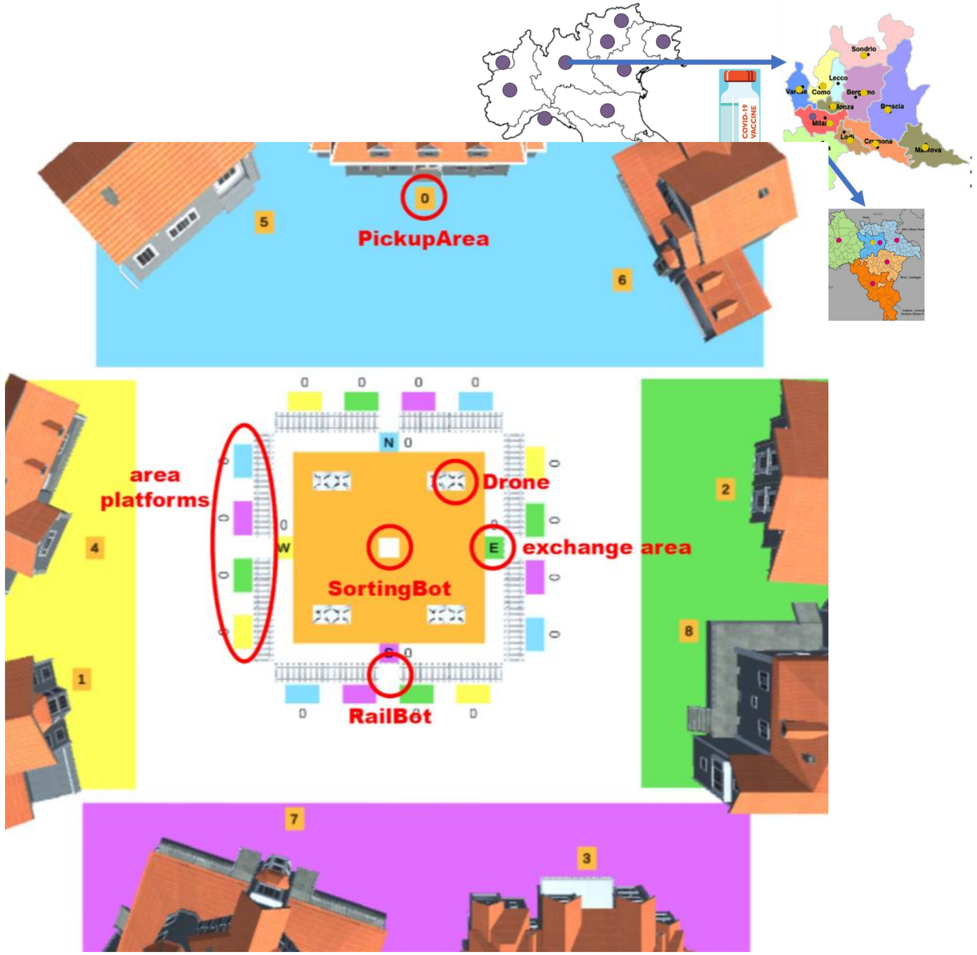
In the previous scenario we had regions, provinces, and health district.

Now, you are provided with a scenario in which:

- Vaccine boxes can be moved between different locations with the use of drones and rails
- Drones are used to pickup and dropdown boxes from the origin and destination locations
- Rails are used to exchange vaccine boxes with the centralized dispatching area
- The Sorting Bot moves boxes between exchange areas

Here is the link to download the Unity project:

https://github.com/2021-unitn-ASA/AOSEProject2020-2021

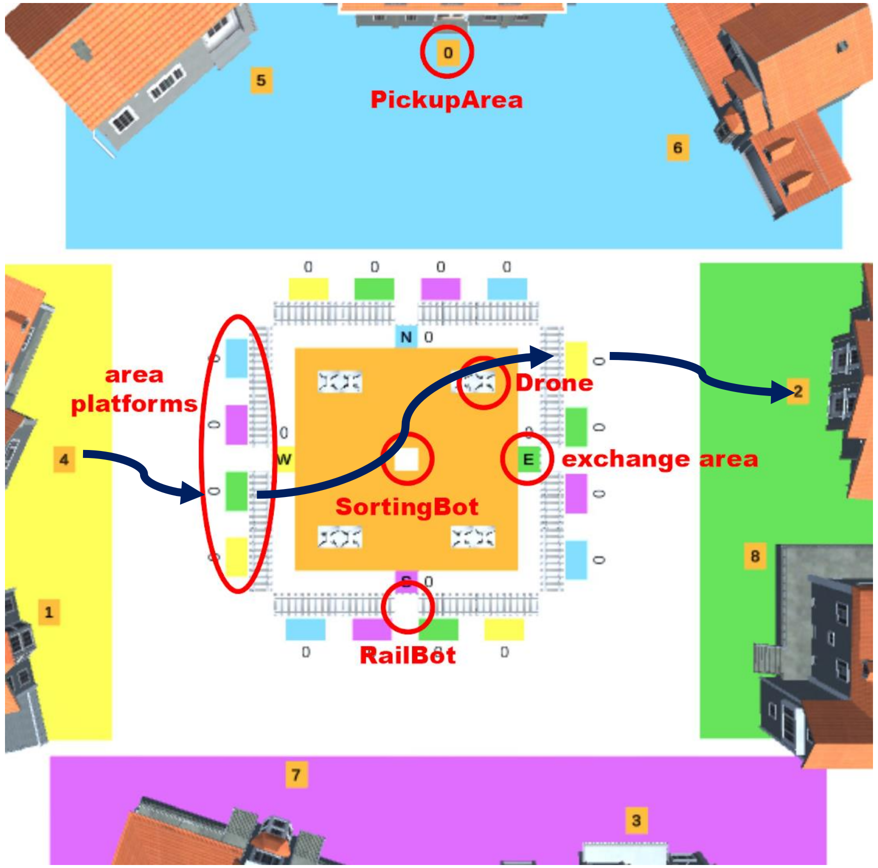


Simulating the vaccine distribution scenario in UnityProlog

The environment

- Each **area** is distinguished by a different color (i.e., north, south, east, west) and labelled on the corresponding exchange area (N, S, E, W). For each area, there are 4 different platforms.
- Each **platform** has the same color of the finishing or starting area, depending whether the box is in the retrieving or delivering phase respectively. The numbers near each platform simply show the number of boxes being there at each instant of time.

For instance, a box that should be delivered from address 4 to address 2, will be transferred by the Drone on the green platform of the west area (because the box has to arrive in the east area, which is green) and then the RailBot of the east area will place the box, given by the SortingBot, on the yellow platform of the east area (because the box has arrived from the west area, which is yellow).



Simulating the vaccine distribution scenario in UnityProlog

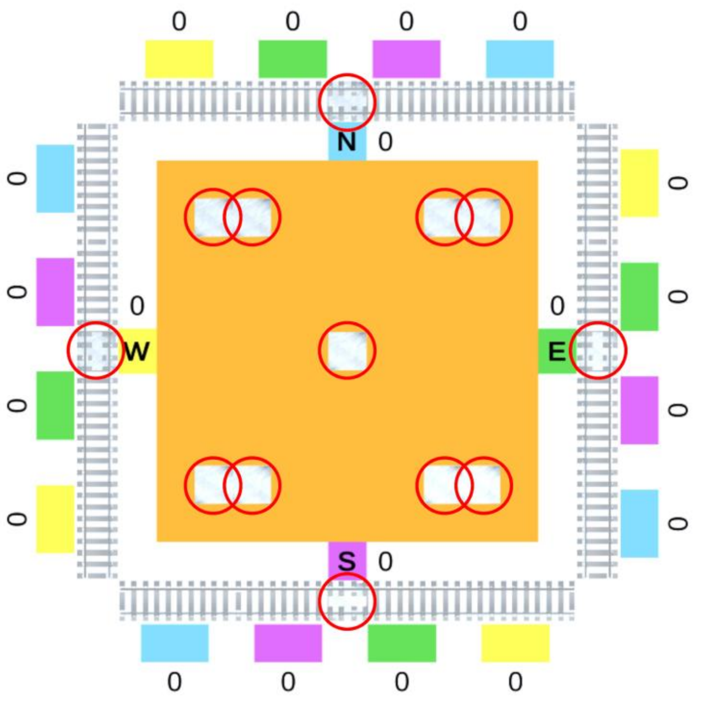
Agents and artifacts

- **Box** represents the model of the boxes that will be instantiated at run-time;
- **PickupArea** These areas are full-fledged agents: they have plans, they can add/remove desires and beliefs from other agents, etc. They have mainly 3 roles:

  1. Represent the post-office box of the house;
  2. Deciding which drone must take care of a box it needs to send;
  3. Recall a box sent to it (actually destroying it).

  As can be seen in the PickupArea.cs code, at the very start of the simulation to each PickupArea a belief is added stating to which area it belongs (i.e., north, south, east, west). This information could be useful to correctly design the knowledge bases of the other agents.

- **GameManager** is the agent responsible for the creation of the boxes. It also triggers the PickupArea representing the address of the sender. We have already provided you an example of implementation for the knowledge base of such an agent, but feel free to edit it in order to make it suit your design choices.



Continue …

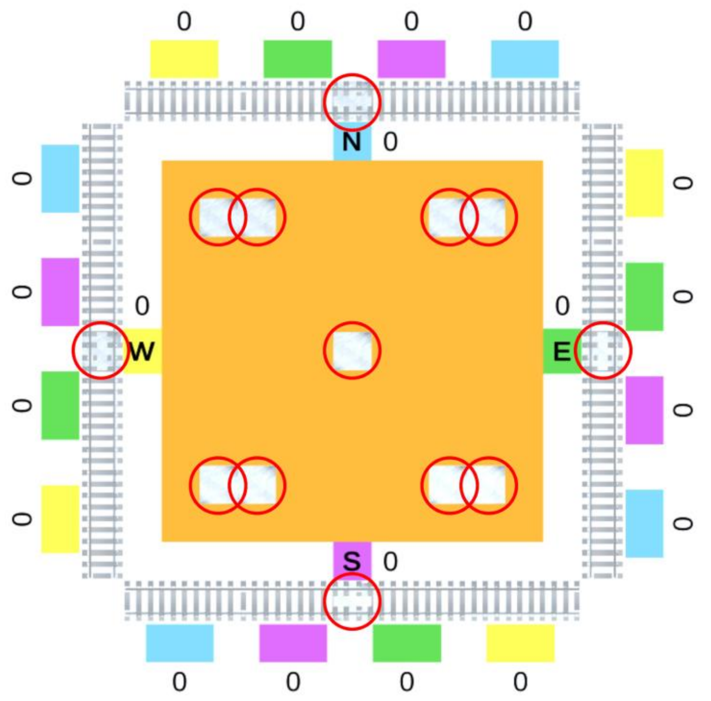
Simulating the vaccine distribution scenario in UnityProlog

Agents and artifacts

…continue

- **Drone** Drones' main responsibility is to retrieve boxes from the PickupAreas and deliver them to the corresponding platform, and vice versa;
- **RailBot** Each RailBot can only move along its own rail. Its main duty is to either { transfer the box from a platform to the exchange area and delegate the management of the box to the SortingBot (if the box must be delivered to a different area), or { leave the box where it is and ask a drone to deliver to the correct address (if box's destination address is in the same area as the starting address);
- **SortingBot** The SortingBot is responsible for the sorting of the boxes from an exchange area to another one. It is invoked by the RailBot of the starting area, and it delegates the management of the box to the RailBot of the destination area.

Drones, RailBots and the SortingBot may not have enough energy to do more than one travel, so they have to recharge after each of them (i.e., have to go back to a charging station before starting a new task). The position of the charging stations is shown by the circles in the Figure.



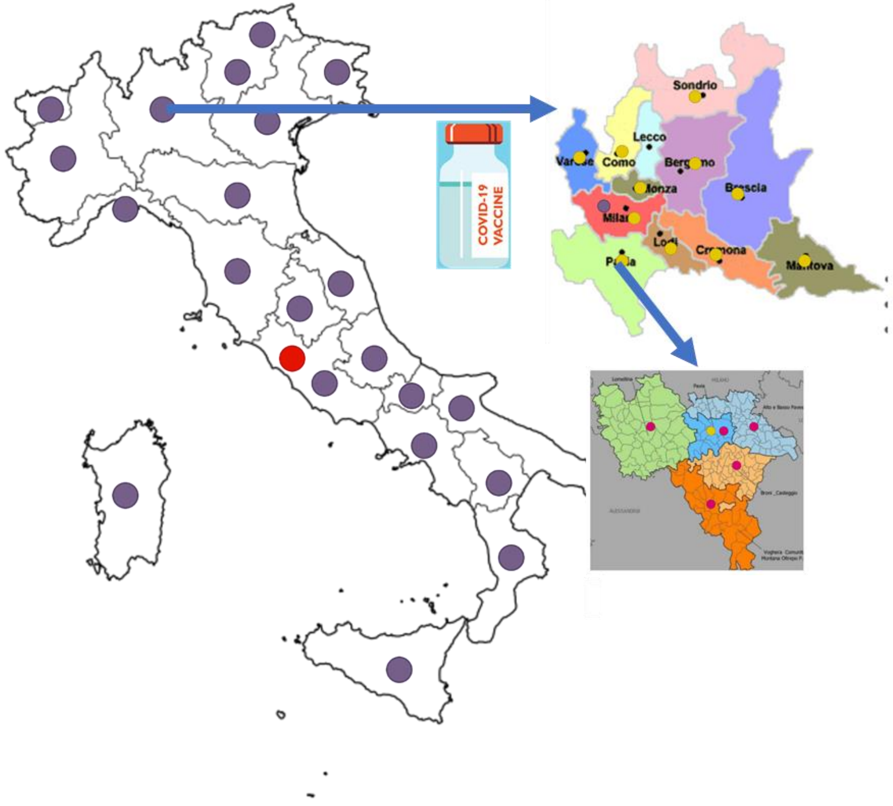
National vaccine distribution scenario in UnityProlog

The Unity project provides entities and C# scripts, but no UnityProlog code. You have to implement your own agents in **UnityProlog**.

- The solution should be completely autonomous, hence there should not be any need for run-time human intervention.
- The implementation has to follow an agent-oriented approach. The delivered knowledge base should work equally well on both your personal test-cases and in our own scenarios; no hard-coded solutions!
- The C# scripts can be modified at will in order to access additional bonus points.
- The project is **individual**, changing the names of the variables of someone else's project will not make it an original work!

A demo video showing the expected behavior is available on the didatticaonline (moodle) platform.

APIs documentation: https://github.com/2021-unitn-ASA/AOSEProject2020-2021



National vaccine distribution scenario in UnityProlog

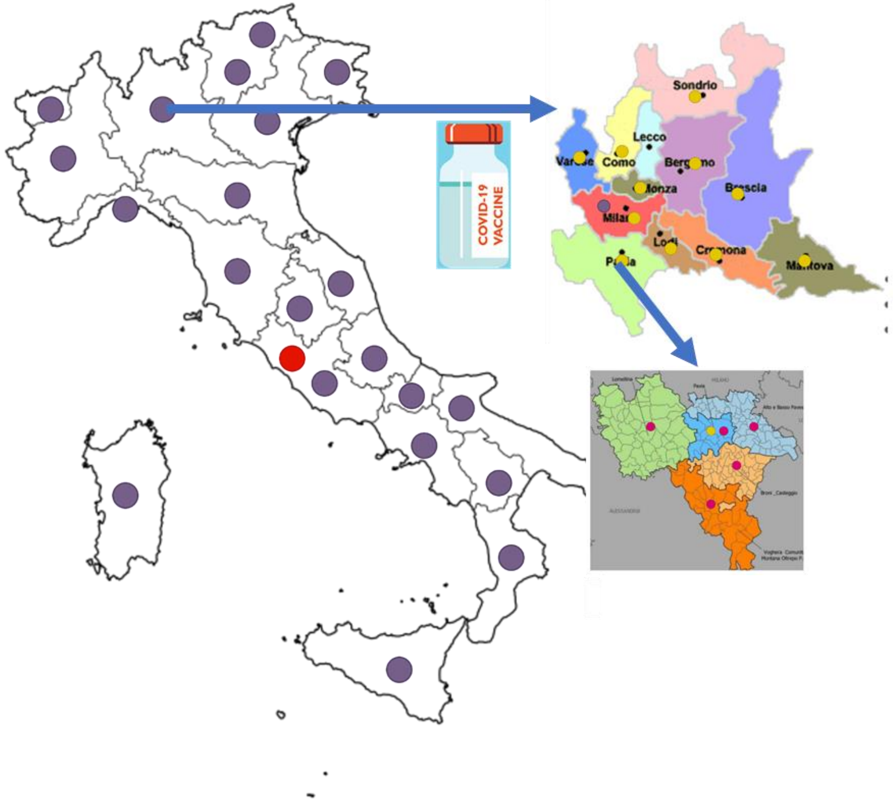
Deliverables

A ZIP archive containing the following files:

1. The “KBs” folder with the .prolog files

2. The “Scripts” folder, only in case you have modified any C# script, or created a new one

3. A PDF report that explains choices taken for the proposed solution, the encountered problems, if any, and how those have been worked around.
   When presenting the solution, also discuss and compare with the national vaccine distribution scenario that you developed in PDDL in the first assignment!



Submission form: https://forms.gle/6igZj8wt6XQGxRe4A

National vaccine distribution scenario in UnityProlog

Deadlines

(13/05) Deadline for submitting A1

...

(25/05) Assignment Unity Prolog (A2)

(09/06) Deadline for submitting A2

(11/06) Notification of results (no oral presentation for the exam)

All students have to register before submitting A2 (9/06)

(02/07 – 10/09) Second and third round

For those not have passed the exam earlier

A1 and A2 have to submitted at least a week before

Submission form:
https://forms.gle/6igZj8wt6XQGxRe4A