CRS Simulator

| Document Reference	2021_IA51_<G3>
Date of publication	2021-03-08
Members of the Group	Carteret thomas.carteret@utbm.fr Iovleff stephane.iovleff@utbm.fr Rothea hadrien.rothea@utbm.fr

CONTENT

. GENERAL PRESENTATION

1.1. GENERAL DESCRIPTION OF THE APPLICATION

The theme of our application is centered around mass public protests. These are very present in France and have a tendency to escalate causing multiple injuries, arrests and damages. Our application will aim to simulate a demonstration with protesters, law enforcement forces along with all the elements that goes with the event. The end goal is to study the behavior of the masses in such scenarios and learn how to minimize damages and maximize civil behavior through the control of the law enforcement forces.

The studied scenario would be a protest march from point A to point B (may add several starting points). The goal of the protesters in a "civil" state would be the travel from A to B. On the other hand, for protesters in an "aggressive" state, they aim to destroy properties around the march. For the ones in a "panicked" state their goal is to run away from the march. As for the law enforcing agents ("players side") the objective is to keep everything under control and stop things from escalating.
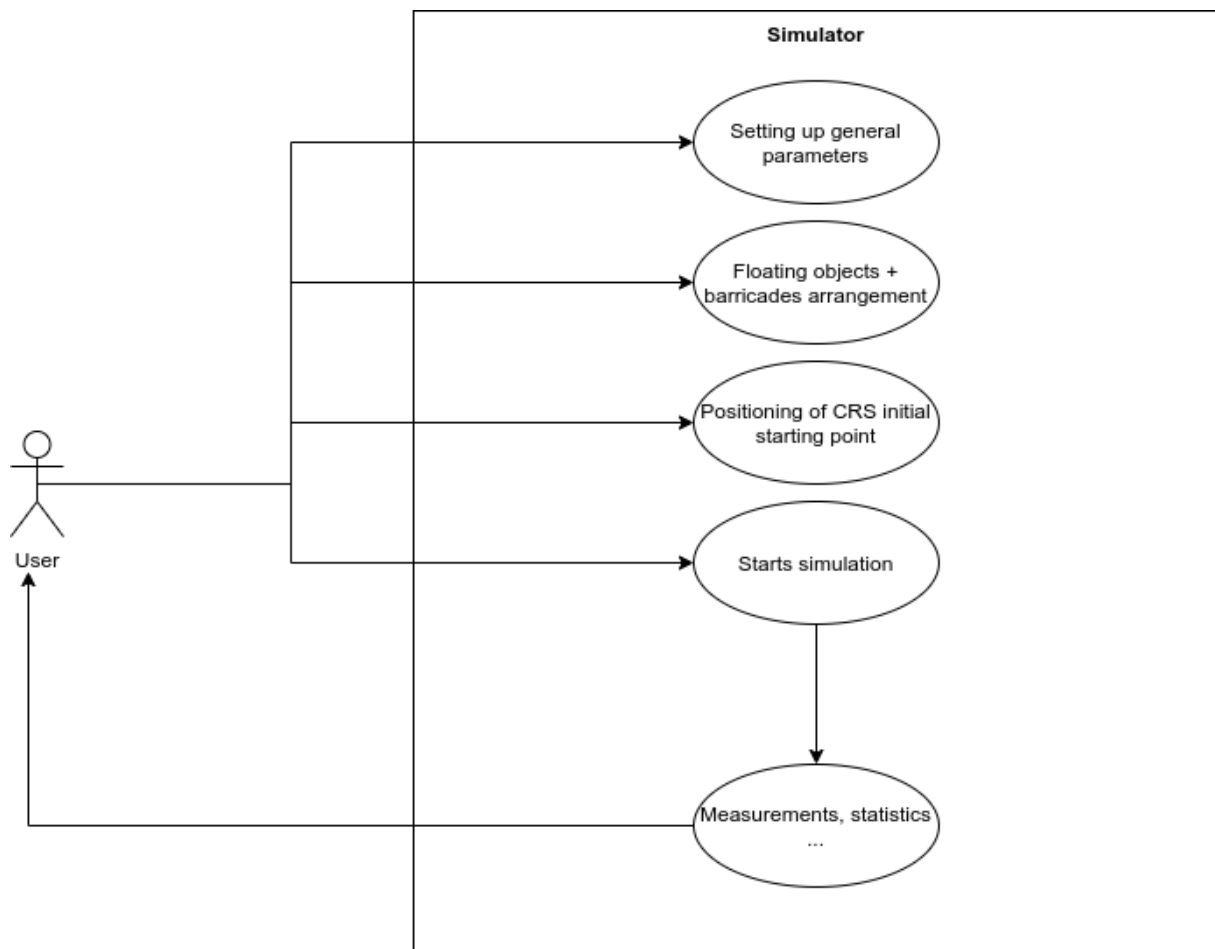
1.2. EXPECTED MEASUREMENTS

To study the behavior of the masses and get estimates on the "civility" of the protest we chose the following measurements:

1. *What:* Damages (in %)
   Why: this will allow us to estimate if the demonstration has degenerated or not.

2. *What:* Evolution of the number of protesters in each state: aggressive, civil or panicked. (in %)
   Why: this will allow us to see the evolution of the protestor's state during the whole demonstration.

3. *What:* Number of people that arrived peacefully
   Why: This serves to show how many protestors ended the demonstration peacefully.

4. *What:* Number of arrests
   Why: This will show the capability of arrest of the CRS and will be an indicator of escalating demonstrations.

5. *What:* Number of escapees
   Why: It will be used to see how much panic was caused during the event.

2. PROJECT MANAGEMENT

2.1. AGILE METHODOLOGY AND SCRUM APPLICATION

We have chosen to use the agile methodology because we are working in small effective with short deadlines. Indeed, this methodology designed according to the startup state of mind which is perfectly suitable for our management requirements. More precisely, we will use the Scrum application and mainly the rituals gravitating around. It will help us to have a dynamic and more human oriented. Our main goal is to avoid the waste of time that the provisional planning that won't be followed represent. The objective here is to focus on efficiency with monthly and weekly objective.

Our adaptation of the Scrum methodology consists of a monthly Sprint Planning in which we define the main needs about the report and the deliverable. The second one is the Weekly in which we define what tasks we embed and what were the difficulties and problems we faced the previous week. To help us managing those events, we use Jira which is an Agile Development tool. Even if, we haven't defined specific roles, Stephane G would be both the Product Owner and the Lead Developer, and we all would be Business Analyst and Developers with our own specializations. Finally, we designated one of us to be the Scrum Master and so the facilitator of the events, he will prepare in advance the meetings to avoid time wasting and fix objectives.

2.2. DEVOPS

Furthermore, we need a day-to-day code release, that is why we will also use the DevOps spirit but only with some of its tools otherwise, it would be a waste of time to deploy every useful tool that would be interesting in a long-term project. For instance, we will use GitHub to manage our code.

3. DETAILLED DESCRIPTION

3.1. ENVIRONMENT DESCRIPTION

To get the desired result on our demonstration control strategies, we need to make an environment that's is as realistic as possible. In order to do that we want to include many environmental elements that will take place in a demonstration. Of course, including all of these elements isn't truly achievable but we can still implement a certain number of objects with the major mechanics that play in such events.

We want to create a basic march demonstration environment where the protestors start the march in various zones and all end in one final zone. The protestors will of course walk on streets and encounter barriers, buildings and so forth.

3.1.1. ENVIRONMENT CONTENT

The simulation contains various elements that can or can't be limited in time and destroyed. We have chosen to class them in two categories according to their time limitation.Persistent content:

➤ **Buildings**: the main goal of this kind of object is limiting the access to the map, we suppose they can't be destroyed.
➤ **Roads**: on the contrary, they are here to lay the path of demonstration, they can't be destroyed. They can also serve as escape routes.
➤ **Floating obstacles**: the main goal of these is to represent elements that can be placed anywhere before the launch of the simulation to observe their impact (in real life, it could be flower pot, urban furniture…), they can't be destroyed.
Barricades: floating obstacles equivalent but for CRS, it can be deployed before and during the simulation and it is destructible.
Destructible content: these objects are destructible and attract the aggressive entities, one of the goals of the police force is to protect those elements. Once it is destroyed, it can be crossed.
➤ **Ending**: this object is not physically present in the environment; it represents the objective for protesters.

Time limited content:

➤ **Smokes:** object of the environment that is not physically present in the simulation but have an impact in the close area.

3.1.2. ENVIRONMENT DYNAMICS

Every element has a different impact on the simulation and so on the agents.

List of the actions in the environment:

- ➢ **LifeDuration:** Will be used to define the lifespan of the smoke.
- ➢ **Destroyed/Damaged:** This action is the result of the aggressive agents attacking destructible contents.
- ➢ **Pull/Push/Block:** Forces exerted by all objects in the environement.
- ➢ **Spawn/despawn:** life cycle of agents at launch.

3.2. INTELLIGENT ENTITIES

3.2.1. PROTESTER

3.2.1.1. EXPECTED BEHAVIOR

Agressive	Civil	Panic
Tolerance A	Tolerance C	Tolerance P

The behavior of the agents is based on their objective as well as the events happening in the environment they can perceive. The behavior of the agent is different for each type. Concerning the protestor this will depend on the state they are in:

- in aggressive mode they will try to break some elements of the environment
- in normal/civil mode they will try to reach a zone set up as their arrival
- in panic mode they will try to flee things they consider as dangerous

3.2.1.2. KEY PROPERTIES

Concerning the protestors, they will have a level of aggressiveness which will decide the state they are in. This level of aggressiveness will be shown as a number between 0 and 100 (still not decided). Moreover, the protestors will have a level of tolerance (randomly generated or decided), this tolerance will make it so that the agent will evolve quicker or slower toward a certain state.

3.2.1.3. INTERACTION WITH OTHER AGENT TYPES

The protestors will be influenced by the actions and behaviors of other agents in their field of view.

Interaction with other protestors: protestors will influence each other's paths and depending on the state of the protestor encountered it may have an effect on the level of aggressiveness.

Interaction with CRS: The CRS will affect the path of the protestors, have an influence on the protestors level of aggressiveness, and will be able to despawn a protestor with a certain amount of aggressiveness.

3.2.1.4. INTERACTION WITH ENVIRONMENT OBJECTS

For the different state of the protestor agent:

- the aggressive one will be able to destroy some elements of the environment
- the normal one will have no interactions with the environment
- the panic one will have no interactions with the environment

3.2.1. CRS

3.2.1.1. EXPECTED BEHAVIOR

The CRS will try to stop aggressive protestors from breaking things.

3.2.1.2. KEY PROPERTIES

The CRS will have no special proprieties, it will only react to aggressive behavior from protestors.

3.2.1.3. INTERACTION WITH OTHER AGENT TYPES

The CRS will be able to arrest the aggressive protestors and also serve as an obstacle in the protestors path.

3.2.1.4. INTERACTION WITH ENVIRONMENT OBJECTS

The CRS will be able to throw objects to scatter people.

4. TECHNOLOGICAL ENVIRONMENT

For this project we made the choice to work with the java and SARL programming languages. We chose to go with SARL for several reasons. First, it's a powerful and diverse tool in mutli-agent programming. It is also quick to set up, conveniently made for group programming and well documented. Of course, we also chose it because we've already started working with this technology.

For the UI we decided to use swing because it is simple to implement and good enough for our application. In this project we truly aim to have a functioning backend simulation more than anything, to really be able to get coherent studies out of our work.

Our project will also use maven to allow the use of libraries like the Arakhnê Foundation Classes with quad tree structures implemented (to be used for the environment).

For the physics of our simulation, we decided to go with the social force model.

5. USER INTERFACE

The interface of this project will be a 2D view of the map from above. On the map the agents will be represented as circles of colors. The design of the objects is still not decided (probably rectangles of colors). The user will be able to choose between different senarios and/or place the number of agents and objects on the map then start the simulation. The user may also be able to choose the different parameters of each agent placed on the map (original state, tolerance, ….).

6. RISK ASSESSMENTS

ID	Type	Description	Mitigation
R1	Methodology	Fully remote project	In this case we must cancel our face-to-face meetings, instead, we will use remote collaborative white boards (like in Teams for instance).
R2	Methodology	A group member doesn't deliver in time or at all	If we face a technical problem, we will try to communicate on it as soon as possible and then try to find help from one of us or the teacher. If the problem is due to a lack of time or desire, we will try to report it in the next weekly and/or change the manager of the task.
R3	Technological	Unoptimized simulation that can't run on every computer	In this situation, we must find a solution to lower the computational needs, we will first search the potentially huge mistakes provoking this overload, then we will try to decrease the needs of the graphic interface.
R4	Technological	A technology used is incompatible with another (Java version for instance)	We will first see if this specific technology is absolutely irreplaceable, if not we will try to change one of the two in conflict. Else, we will try to find a gate between the technologies. The objective would be to detect that kind of problem as soon as possible.
R5	Management	A group member gets sick	We will try as much as possible to transfer its work to the two group members remaining.
R6	Management	A conflict between two group members about who will be in charge of certain tasks.	If both members have the same amount of work and the same abilities to achieve the task, we will decide with one shot rock-paper-scissors.