



Institut de Recherche  
pour le Développement  
FRANCE

French National Research Institute for Sustainable Development



## SESSION 2A – PRESENTATION OF GAMA

Alexis Drogoul, UMMISCO, IRD  
Patrick Taillandier, UMMISCO, INRAE  
Arthur Brugi  re, UMMISCO, IRD





# GAMA, a platform dedicated to build spatially explicit agent-based models and run simulations

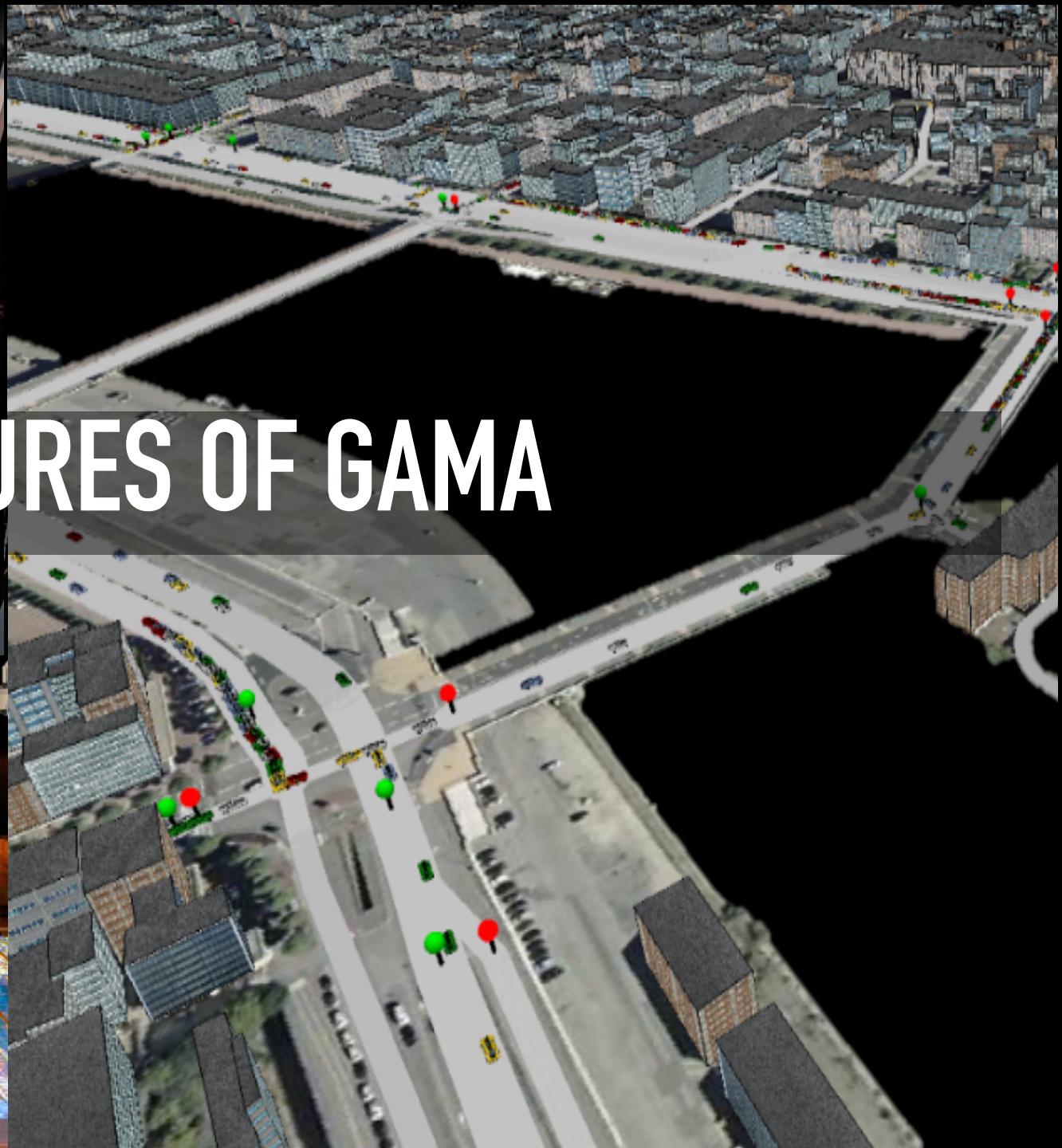
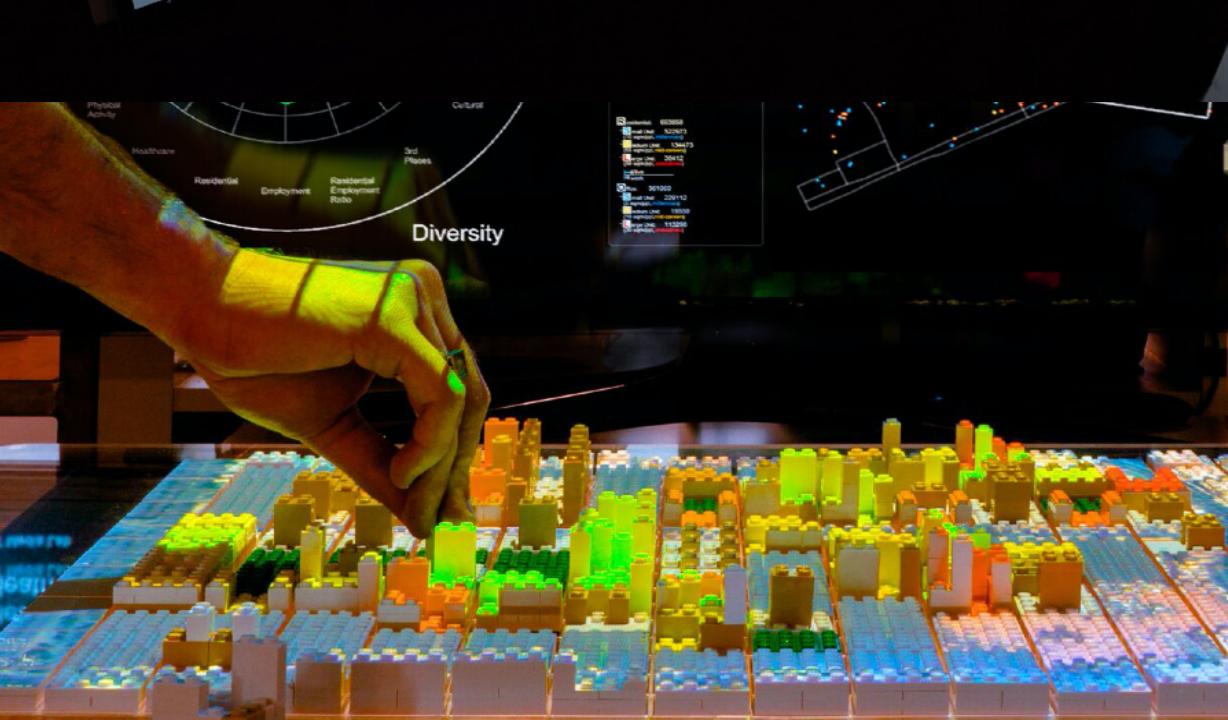
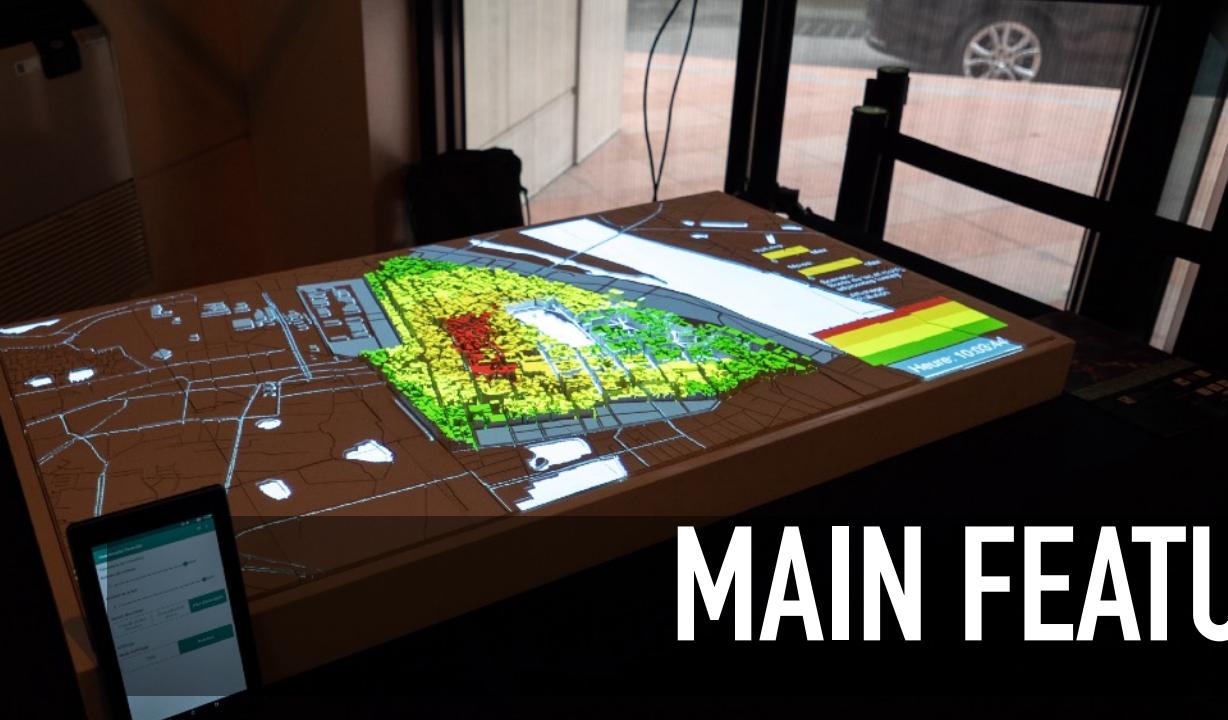
- Developed by a consortium of research team : UMMISCO (IRD - coordinator), ACROSS (Hanoi), DREAMS (Can Tho), UT1-IRIT (Toulouse), INRAE-MIAT (Toulouse), LRI (Orsay), MIT Media Lab (Boston)...
- **Generic**: can be used for a wide range of applications
- Developed under GPL license: **open-source** (GNU v3)
- Allows **modelers** (even non computer-scientists) to build complex models **quickly and easily**:
  - Integrates a complete modeling language (**GAML**) and
  - an **Integrated Development Environment**
- Developed in **JAVA** : **easy to extend** in order to take specific needs into account

# GAMA overview: strength of GAMA



- Powerful modeling language with a simple syntax and numerous built-in operators
- Modern IDE
- Seamless integration of geographic data and GIS tools with agent-based models
- Supports the development of complex models with thousands of agents and rich behaviors
- Integrates powerful visualisation tools

# MAIN FEATURES OF GAMA

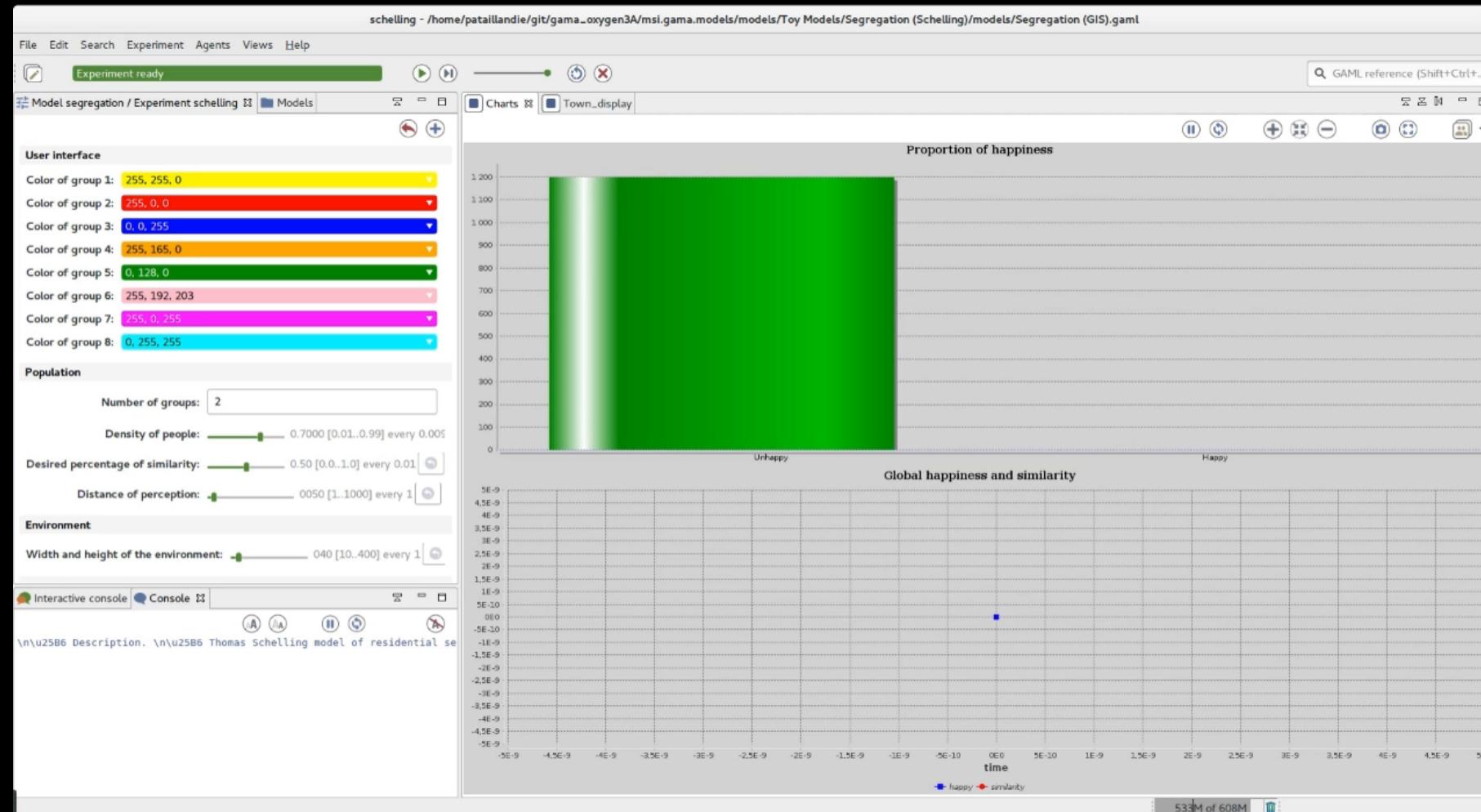


# GAMA provides a complete Integrated Development environment (IDE) to build models



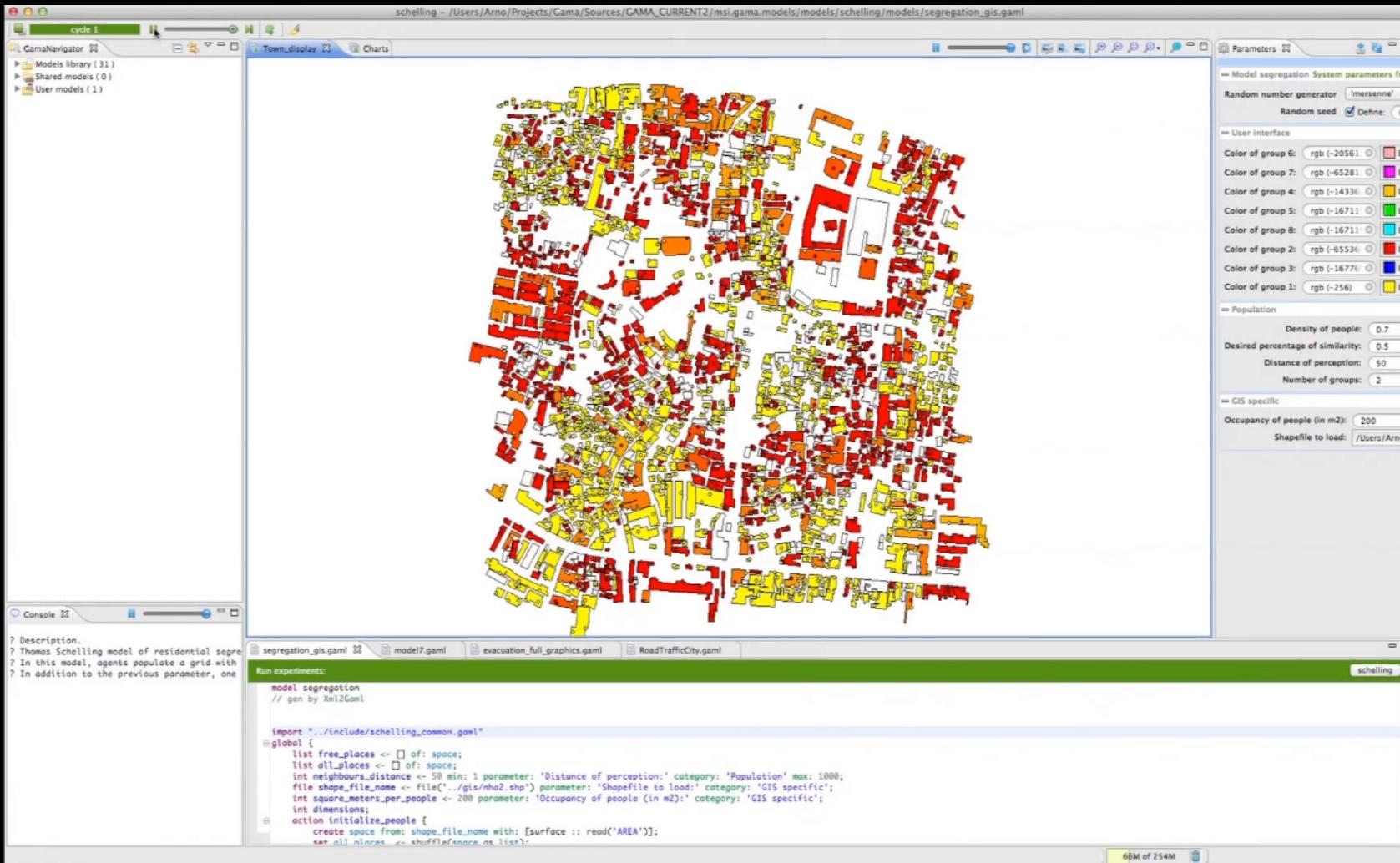
Dedicated modeling Language (GAML), easy to learn and to extend

# GAMA provides a modern and flexible modeling and simulation interface



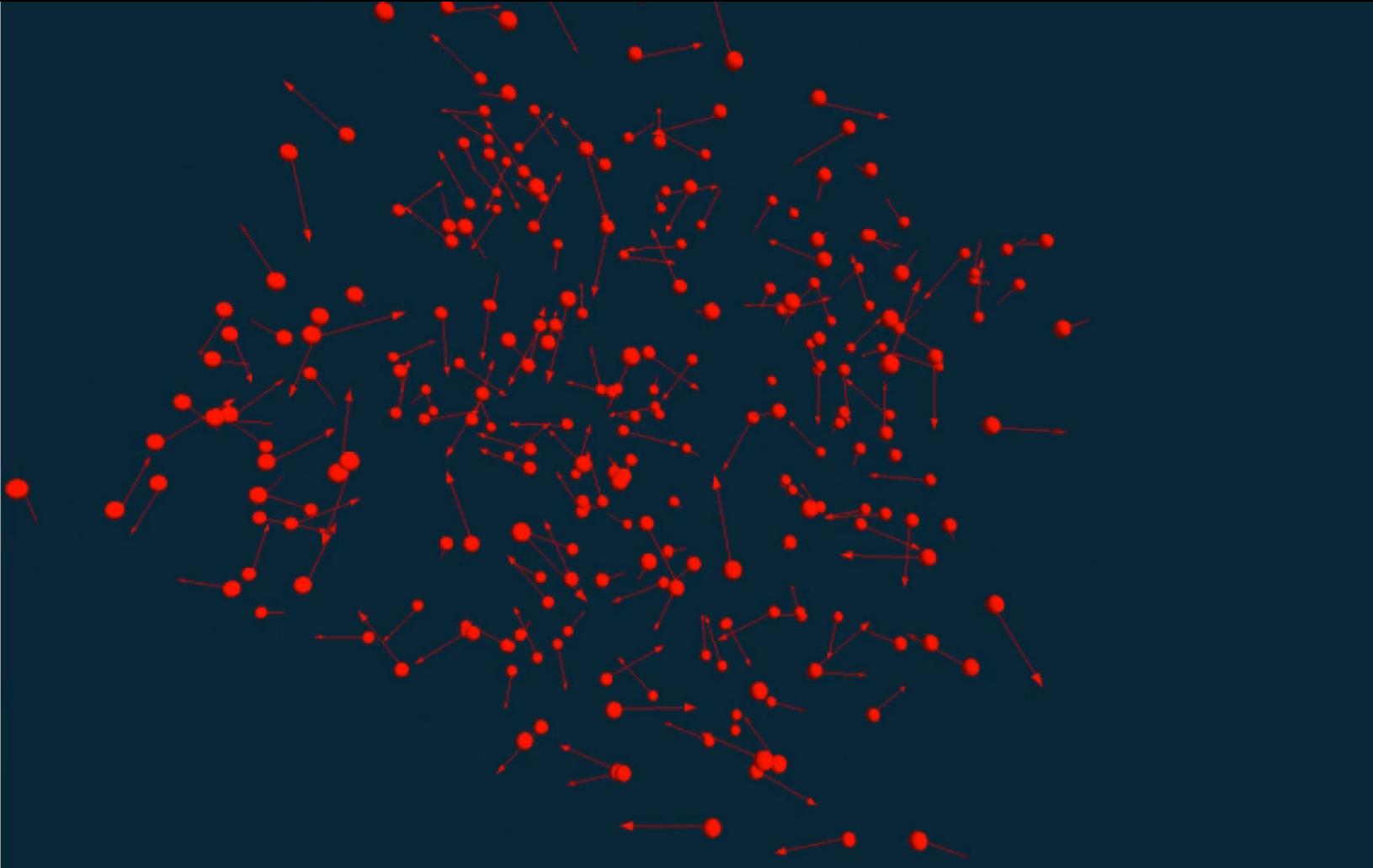
Panel control, agent inspector and browser, data viewers, integrated documentation

# Easy integration of GIS data, powerful features to manage GIS data (many spatial operators)



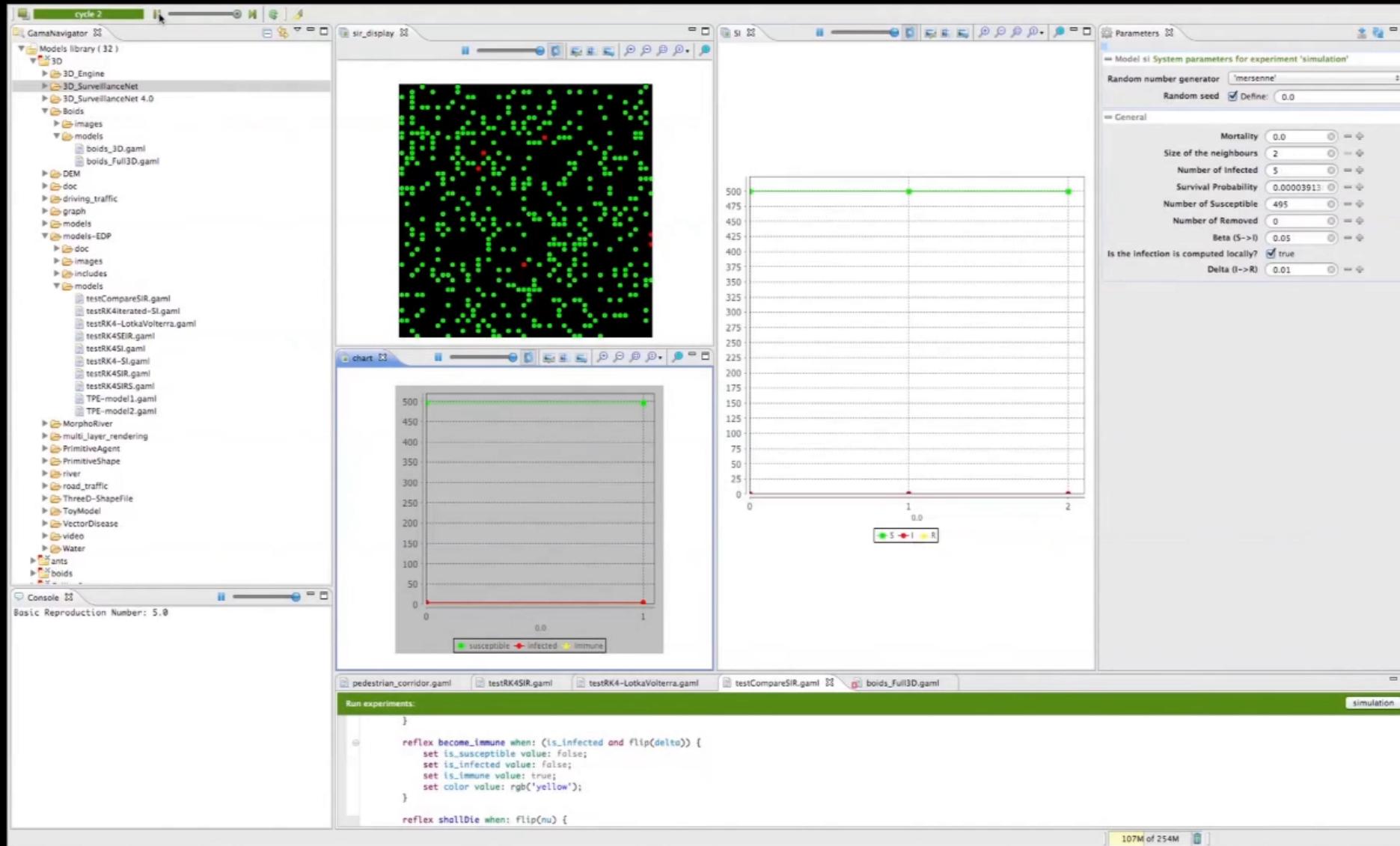
Transparent agentification of 2D/3D GIS data (shapefiles, OSM, asc) powerful geometrical operations (union, difference, spatial queries....) , export into Shapefile/asc/geotiff/KML

Many optimized operators dedicated to agent movement and perception on continuous, grid and graph topologies



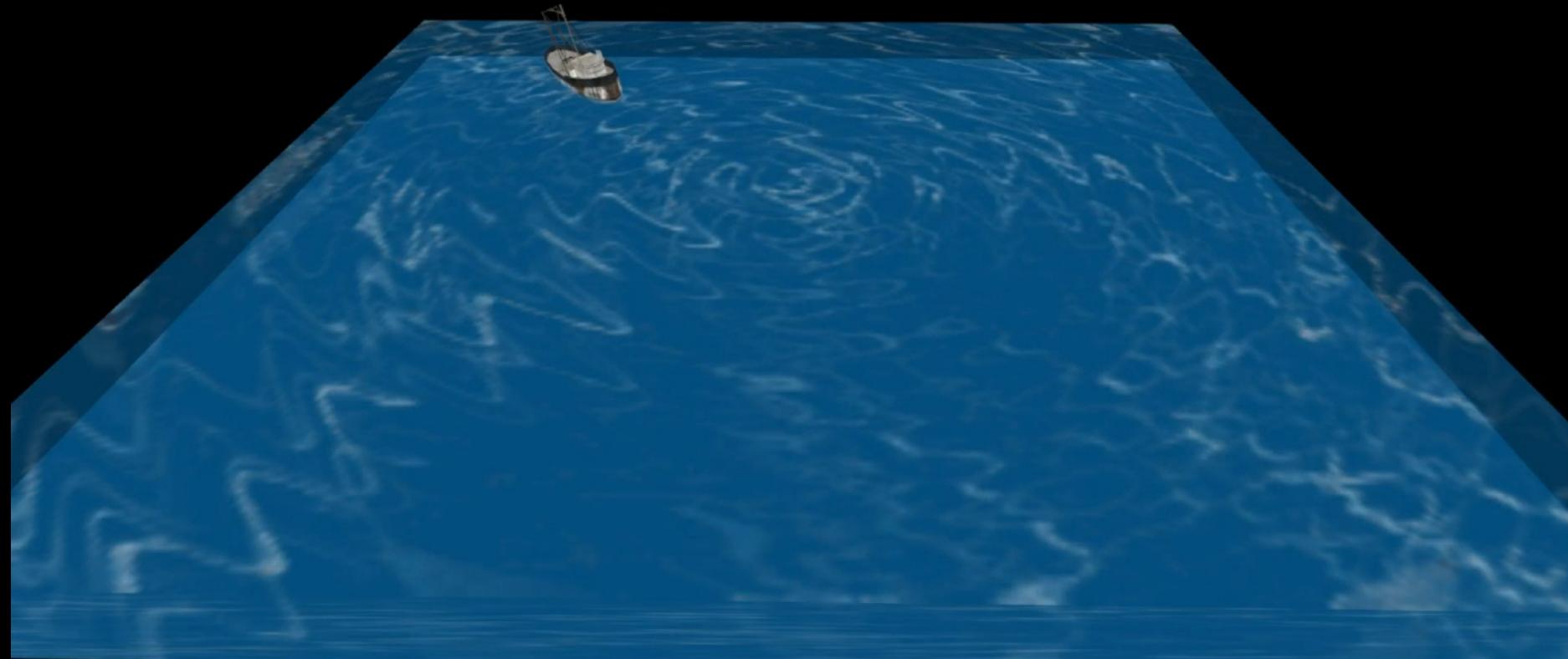
Many integrated algorithms: Dijkstra, Floyd Warshall, A\*, NBA\*, JSP (grid)...

# Allows to use different formalisms to define agent behaviors



Differential equations, BDI architecture, Finite state machine, Reflexes, ....

Powerful visualization tools allowing to define as many displays as necessary

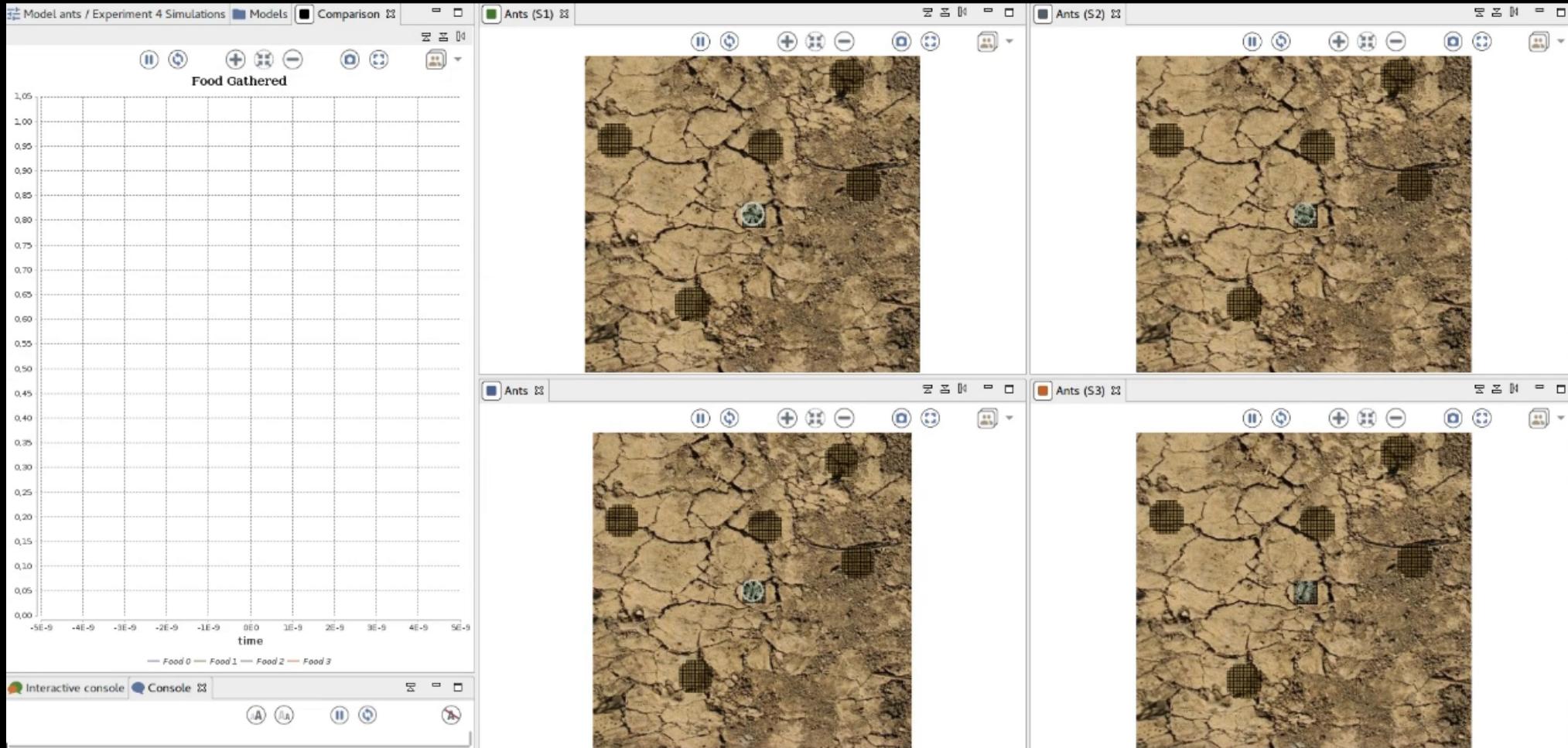


3D display, use of layers, Chart Display....

# Powerful user-interaction tools allowing to interact with simulations

Event layers, user commands, simulation interconnection

# Advanced features: multi-simulation, multi-level, co-modeling



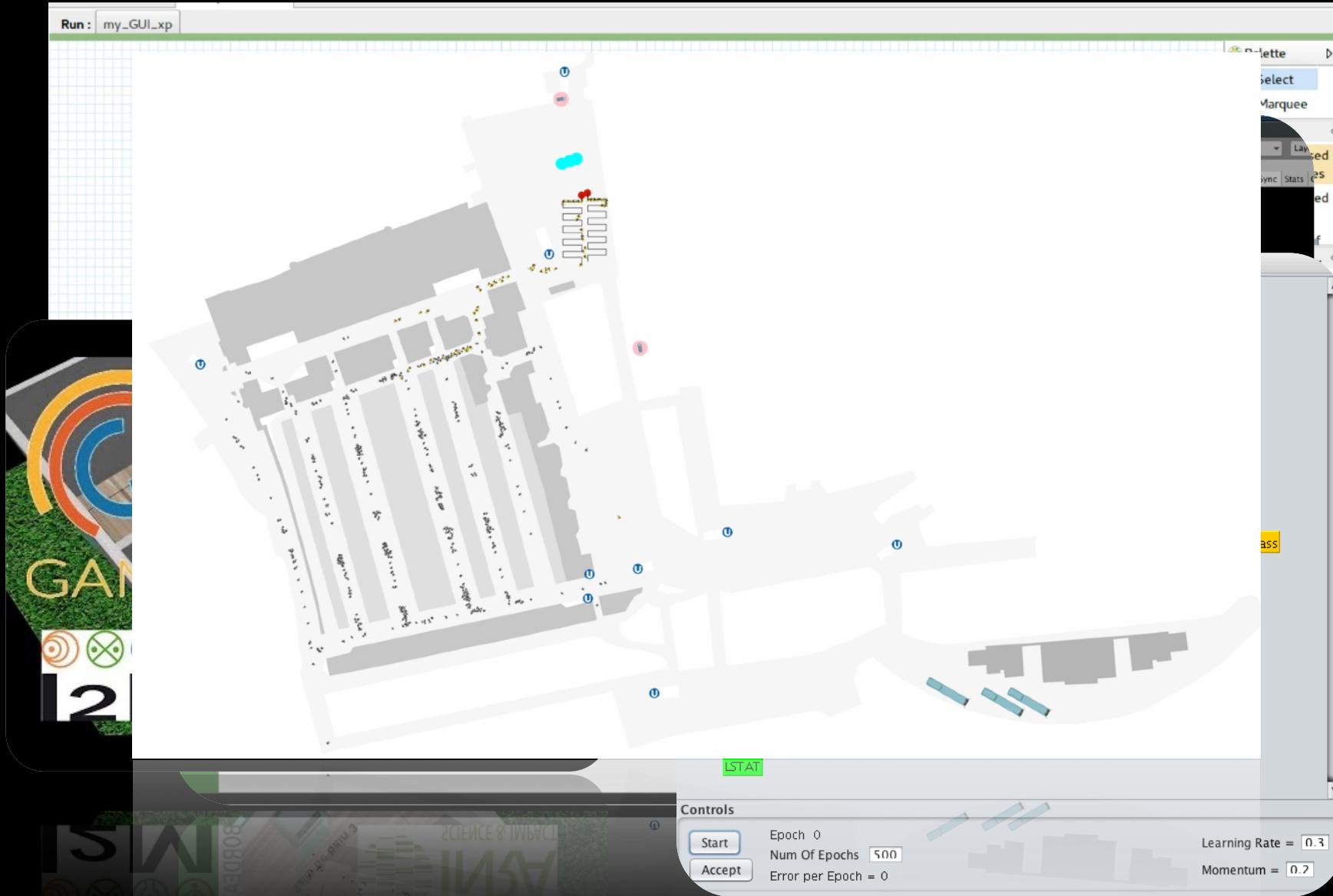
# Model exploration and calibration

The image displays a composite of several elements:

- A large blue "R" logo, which is the symbol for the R programming language.
- An OpenMole graphical user interface window titled "Parameters". It shows parameters for an experiment named "Genetic".
  - Random number generator: "mersenne" (selected from a dropdown menu "among [cellular, xor, java, mersenne]").
  - Random seed: 0.0 (with a "Define:" checkbox checked).
  - Exploration method settings:
    - Stop condition: time > 1000.
    - Best fitness: 1.7976931348623157E308.
    - Last fitness: 299.0.
    - Parameter space: infection\_rate(6) \* speed\_people(10) = 60.
    - Exploration method: Method genetic | fitness = minimize nb\_infected | compute the min of 3 simulations for each solution.
    - Mutation probability: 0.1.
    - Crossover probability: 0.7.
    - Population dimension: 3.
    - Preliminary number of generations: 1.0.
    - Max. number of generations: 5.0.
  - Parameters to explore:
    - Infection rate: 0.8 (selected from a dropdown menu "among [0.1, 0.2, 0.5, 0.6, 0.8, 1.0]").
    - Speed of people: 5.0.
- Two cartoon illustrations:
  - A mouse wearing a green hat and a circular badge that says "RC 1". Above the mouse, the text reads "Invisible Init" and "2.". Below the mouse is a "Download" button.
  - A pink figure wearing a crown and a yellow "4" on its chest, standing inside a dark green oval. Below the figure is the text "Killer King".

Integrated batch and Headless mode, connection with Python, R and OpenMole

# Many extensions: graphical modeling, Unity 3D, machine learning....



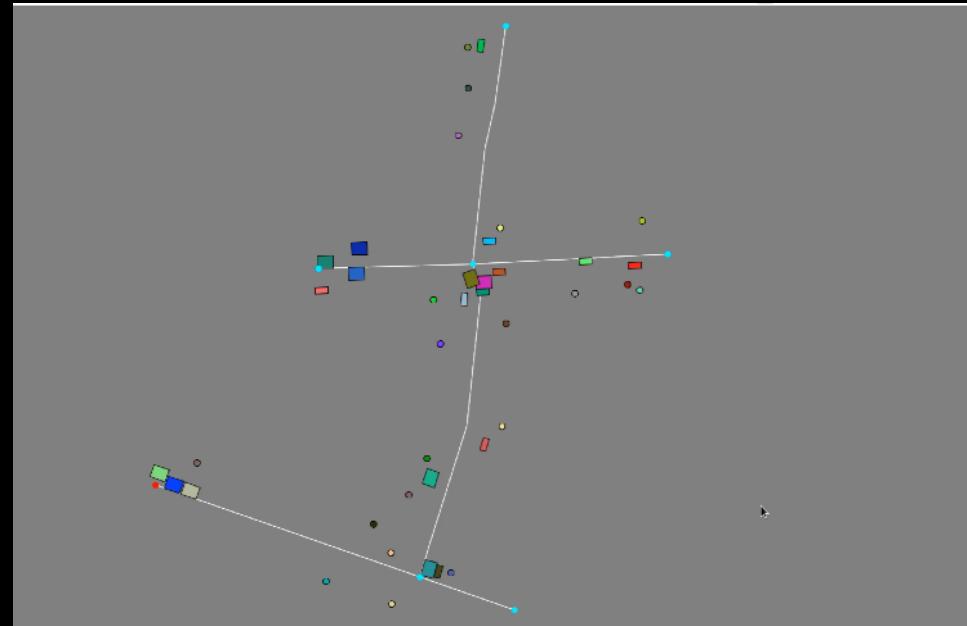
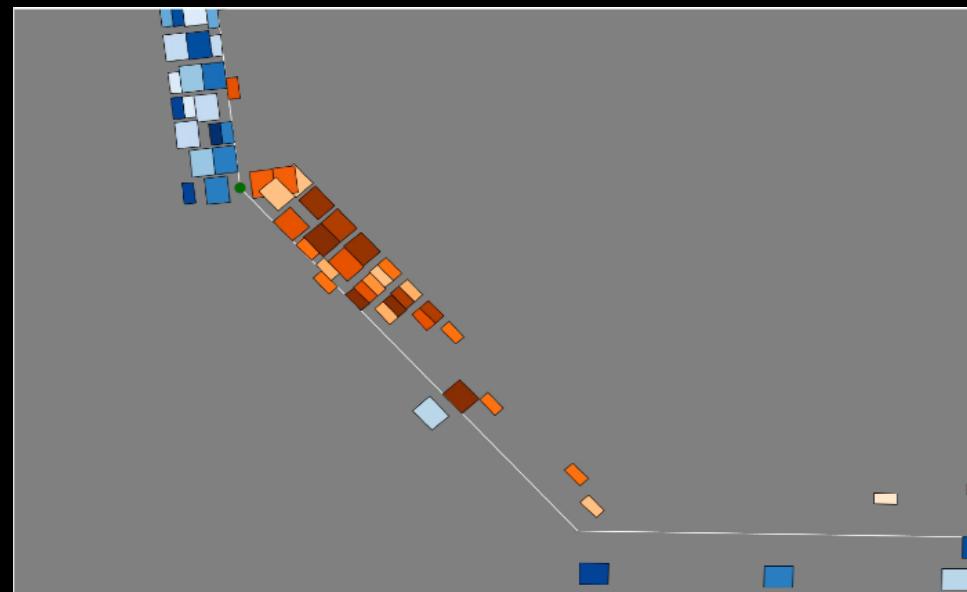


# EXAMPLE OF APPLICATION: MOBILITIES & AIR POLLUTION

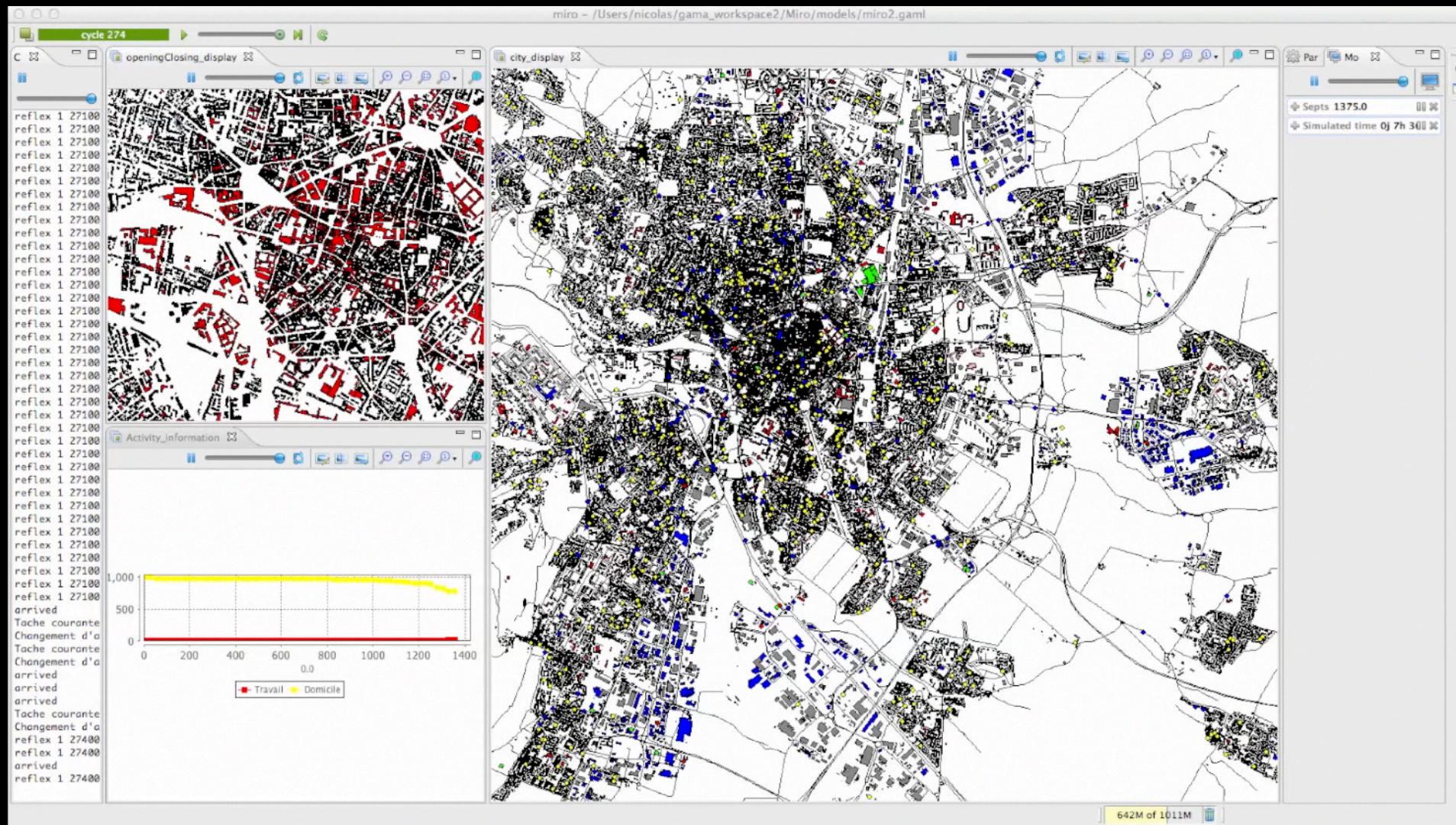
# SIMULATING THE TRAFFIC IN VIETNAM

Traffic in HCM/Hanoi (data from 2017) :

- ✓ 74% de motos
- ✓ 19% de vélos
- ✓ 6% de piétons
- ✓ 1% de voitures



# MIRO2 PROJECT: how to improve individual accessibility to the city in order to better manage urban mobility?

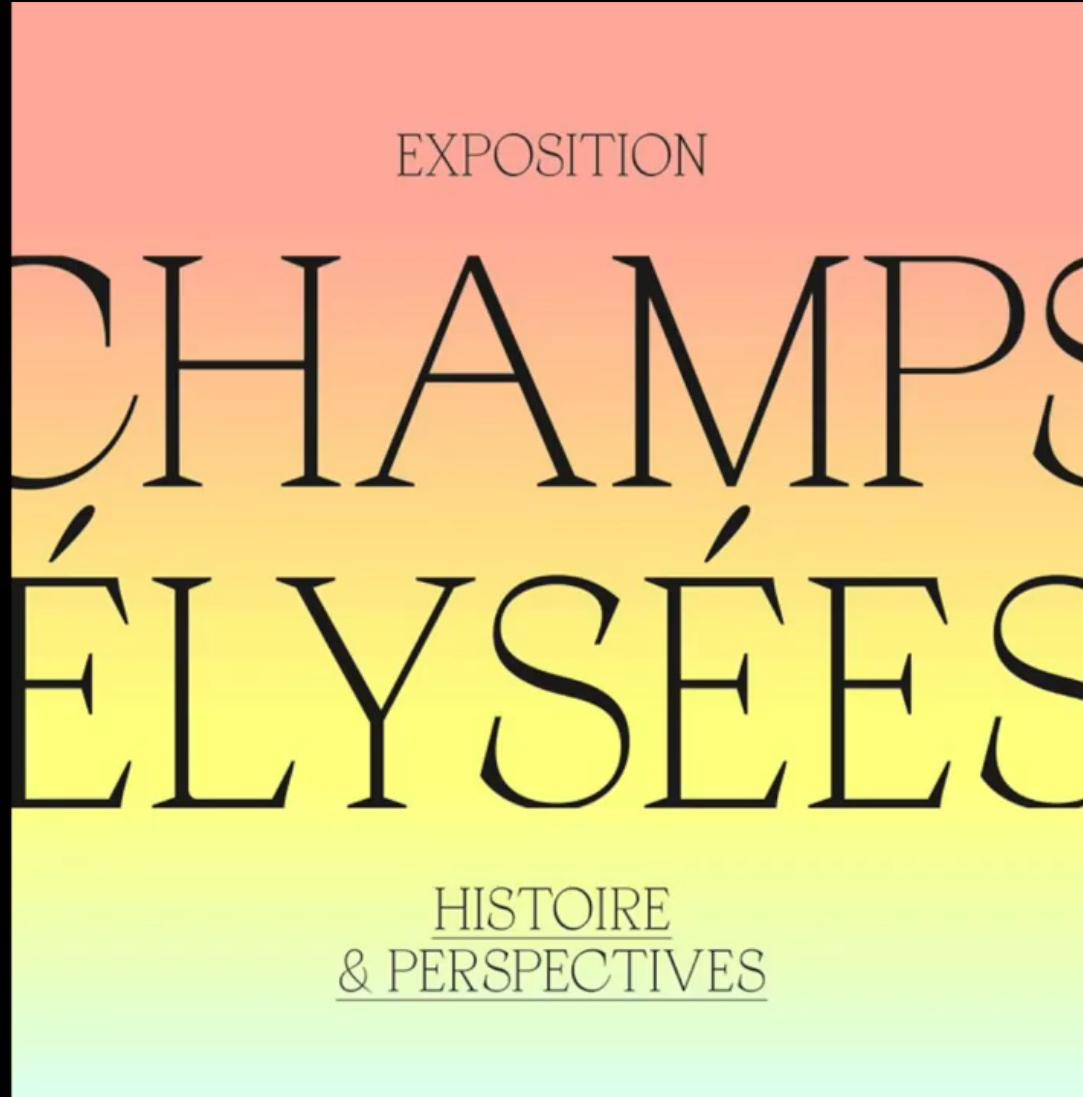




## EXAMPLE OF APPLICATION: CITY DESIGN



# CITYSCOPE CHAMPS-ELYSEES (PARIS, FRANCE)

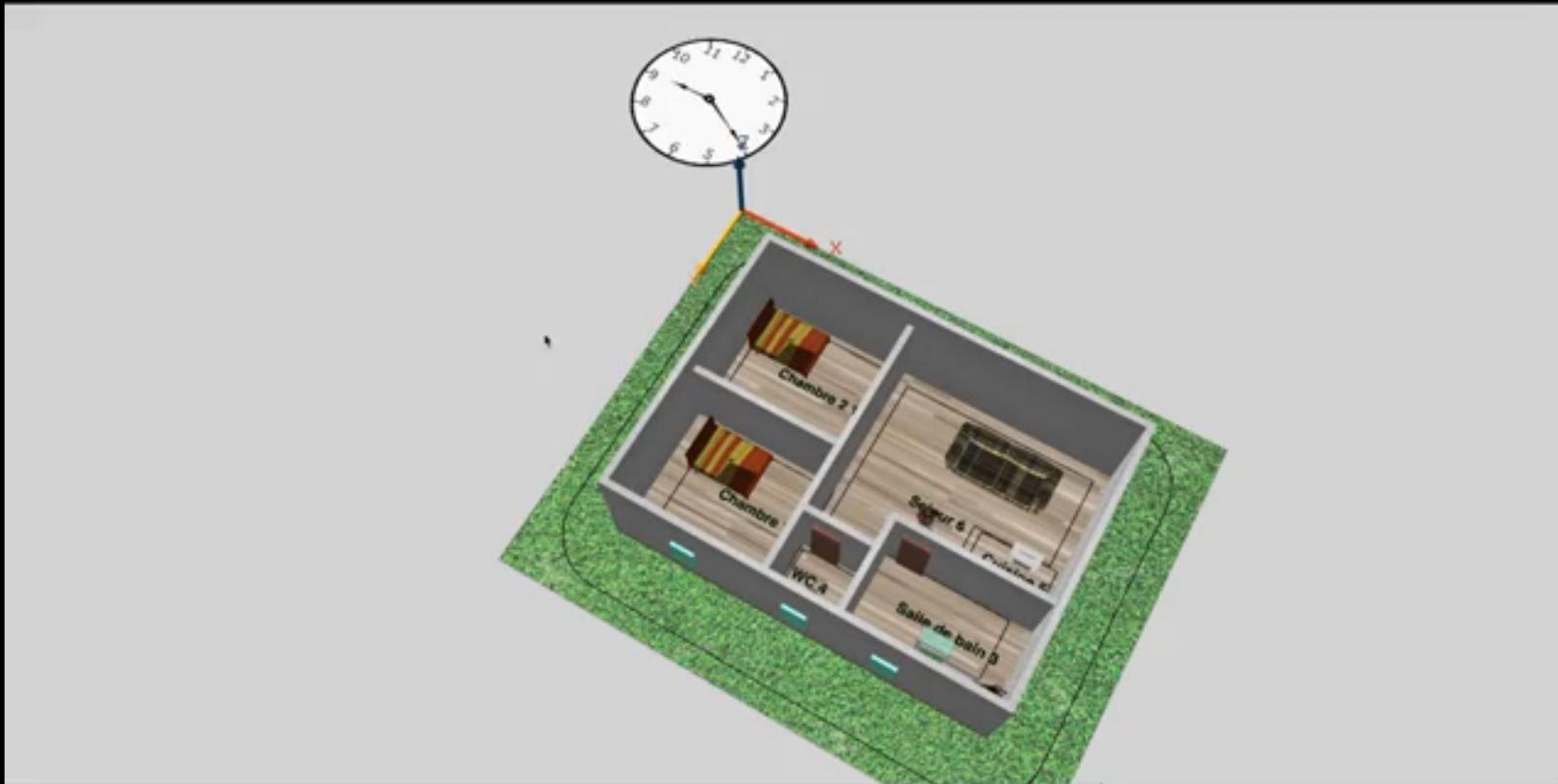


# CITYSCOPE VOLPE



Alonso, L., Zhang, Y. R., Grignard, A., Noyman, A., Sakai, Y., ElKatsha, M., ... & Larson, K. (2018, July). Cityscope: a data-driven interactive simulation tool for urban design. Use case volpe. In *International conference on complex systems* (pp. 253-261). Springer, Cham.

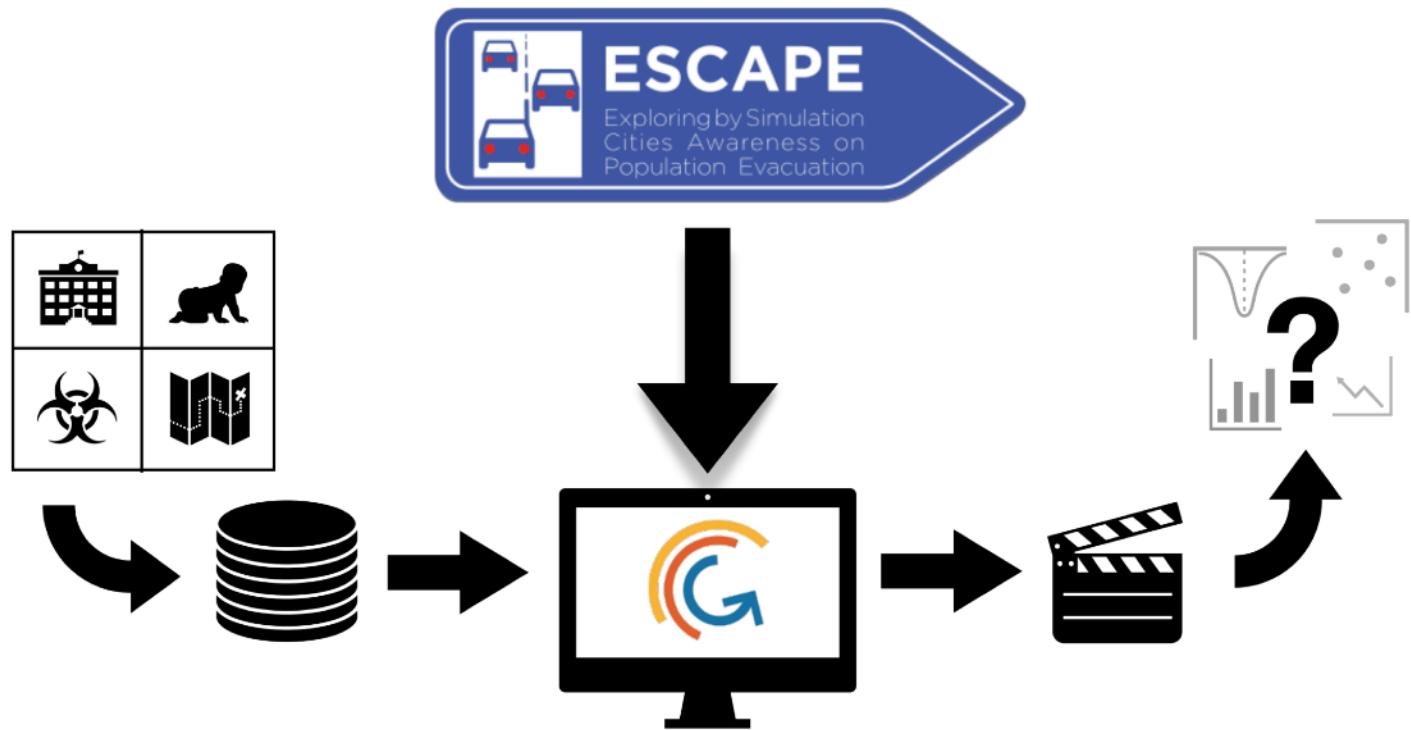
# Li-BIM: Simulation of the behavior of the occupants in a building and their indoor comfort



Micolier, A., Taillandier, F., Taillandier, P., & Bos, F. (2019). Li-BIM, an agent-based approach to simulate occupant-building interaction from the Building-Information Modelling. *Engineering Applications of Artificial Intelligence*, 82, 44-59.

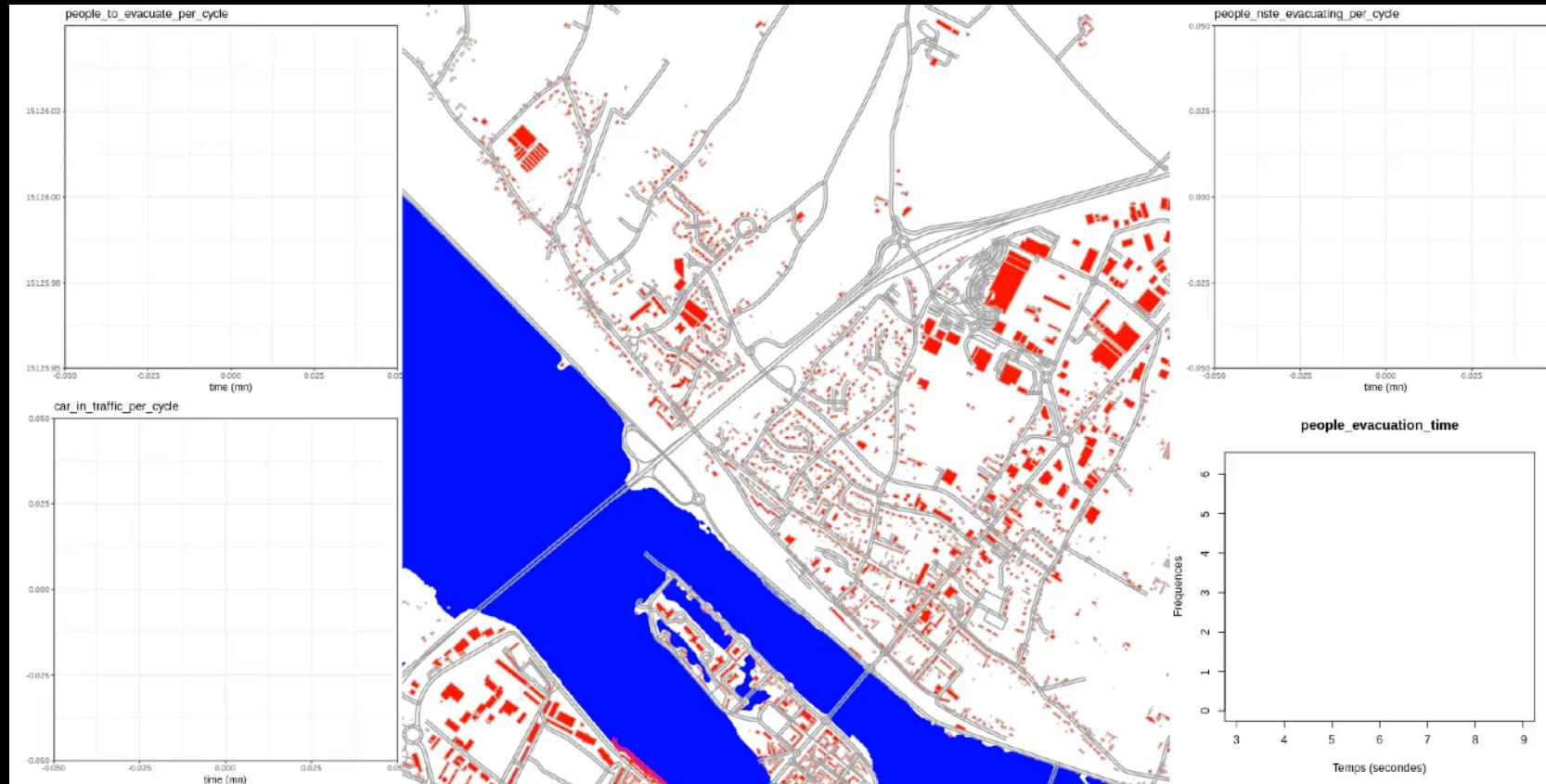
# EXAMPLE OF APPLICATION: RISK MANAGEMENT





**Objective:** test prevention  
plans through simulation

# APPLICATION EXAMPLE: SAUMUR (France)

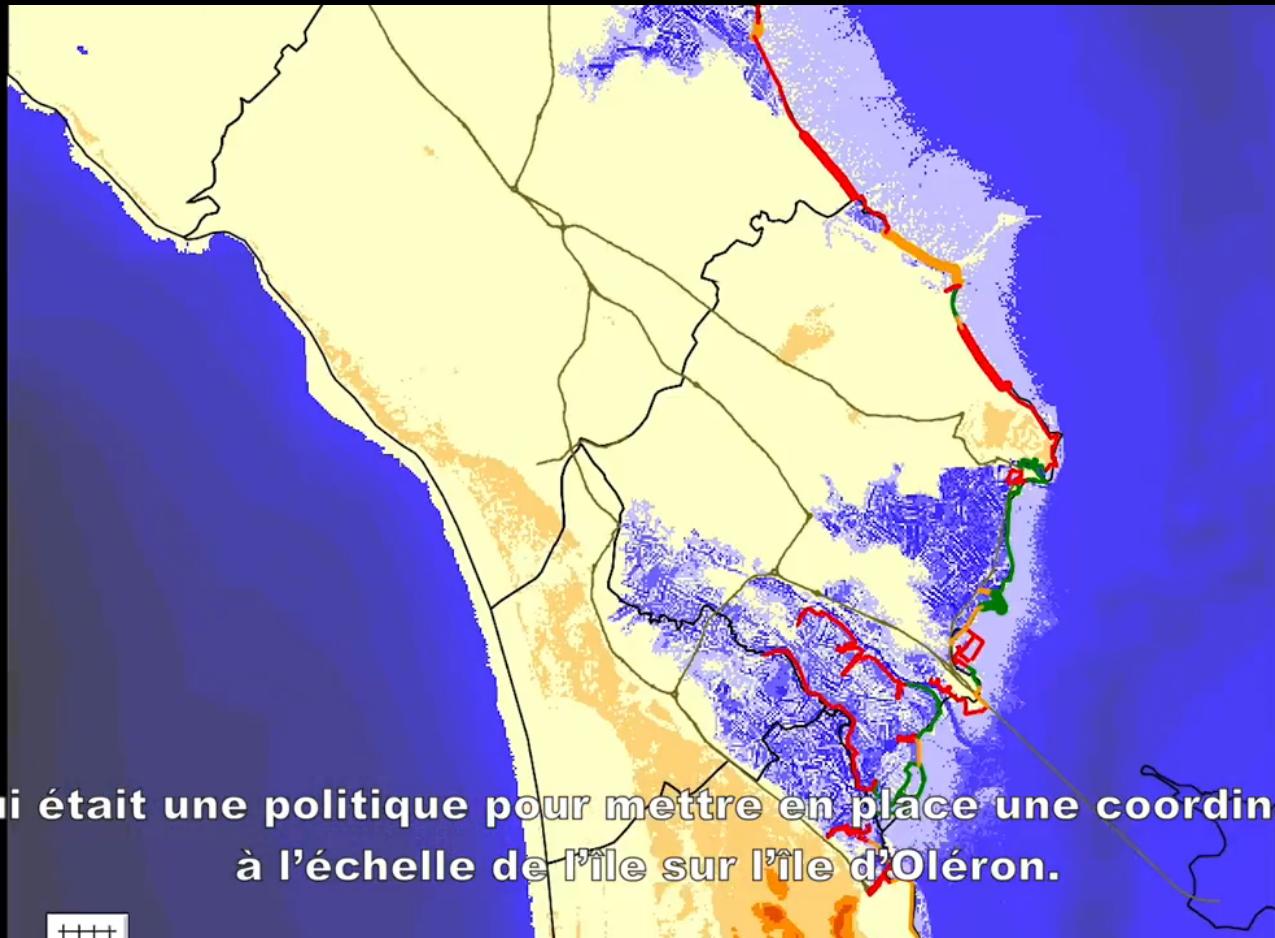


Flooding of the Loire river with the sudden formation of a breach in the dike on the right bank, protecting part of the city of Saumur and neighboring towns located at the foot of the hillside (Authion valley)

# LITTOsim: A MULTI-PLAYER SERIOUS GAME

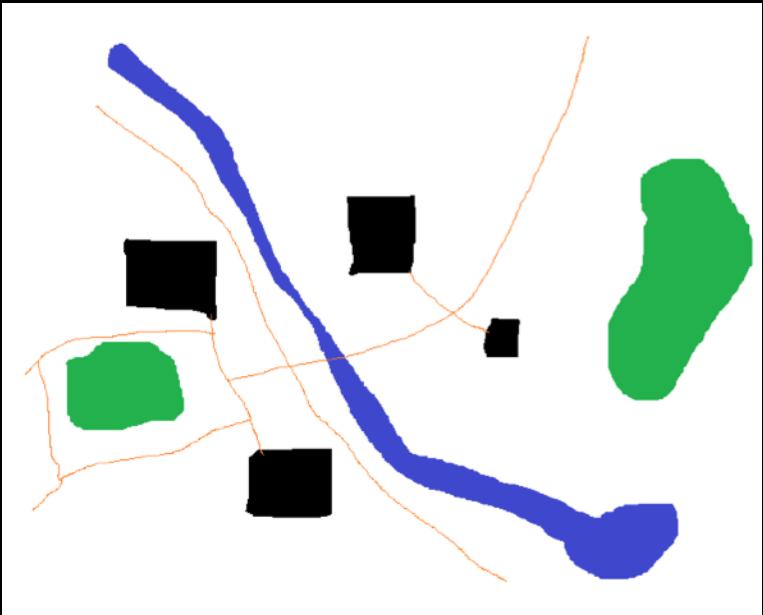
LITTOsim :

Participatory simulation of  
coastal flooding - building social  
learning on prevention  
measures  
with decision-makers

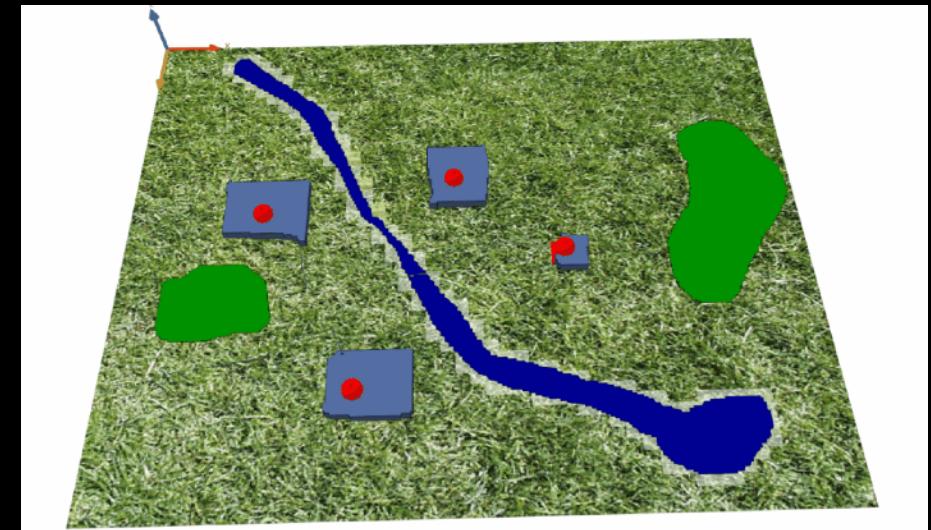


# DYCOMA: THE DYNAMIC MIND MAP AS AN AID TO REFLECTION ON FLOODING IN THE CITY

**Objectives** : allow to understand the participants' representation of the territory, to put it in comparison with the flood risk and allows them to think about the impact of urban planning choices and flood management strategies (grey solutions, nature-based solutions...).

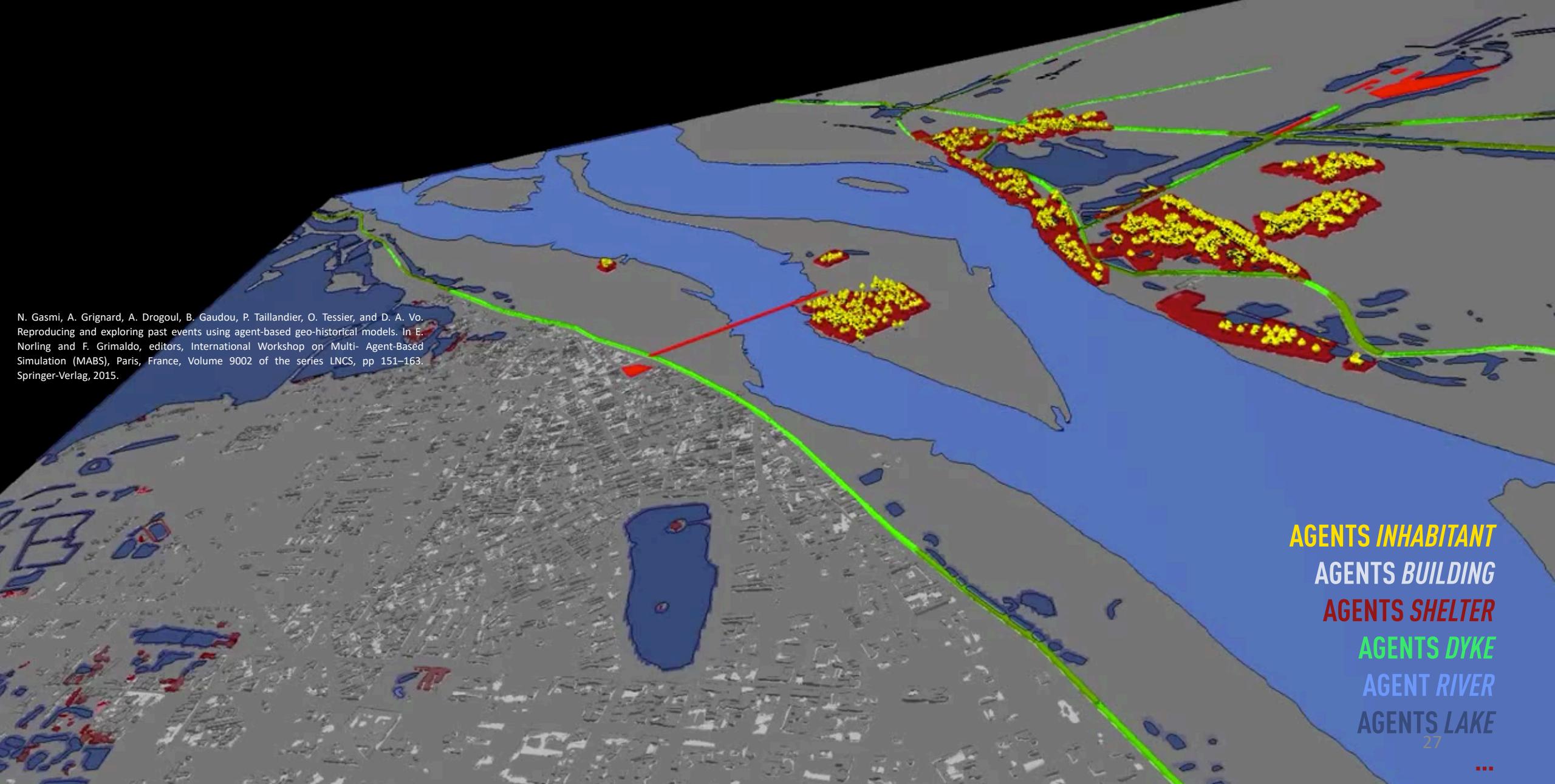


We ask people to draw their environment



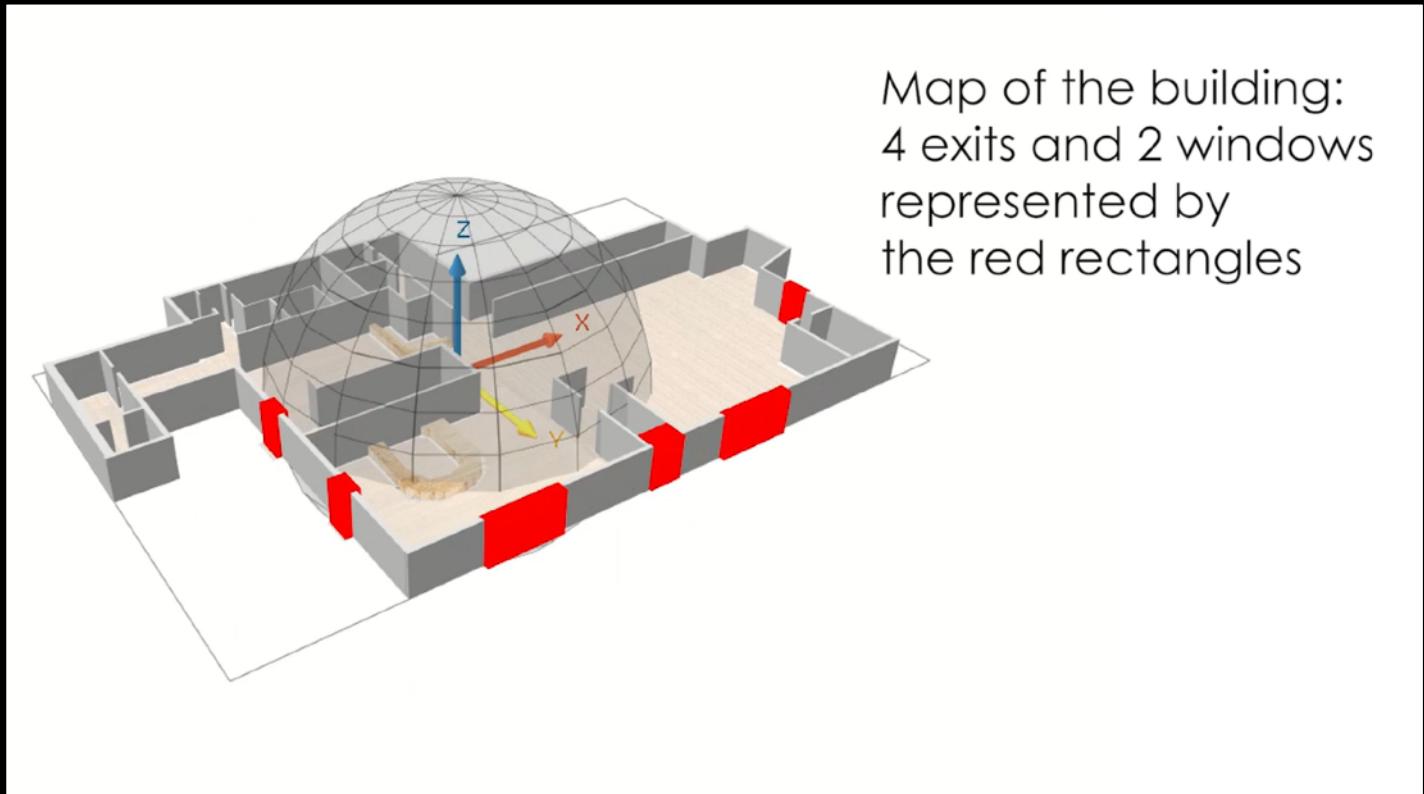
From the digitalized version of the drawing, we simulate a flood

# ARCHIVE PROJECT: REPRODUCTION OF THE 1926 FLOODS IN HANOI

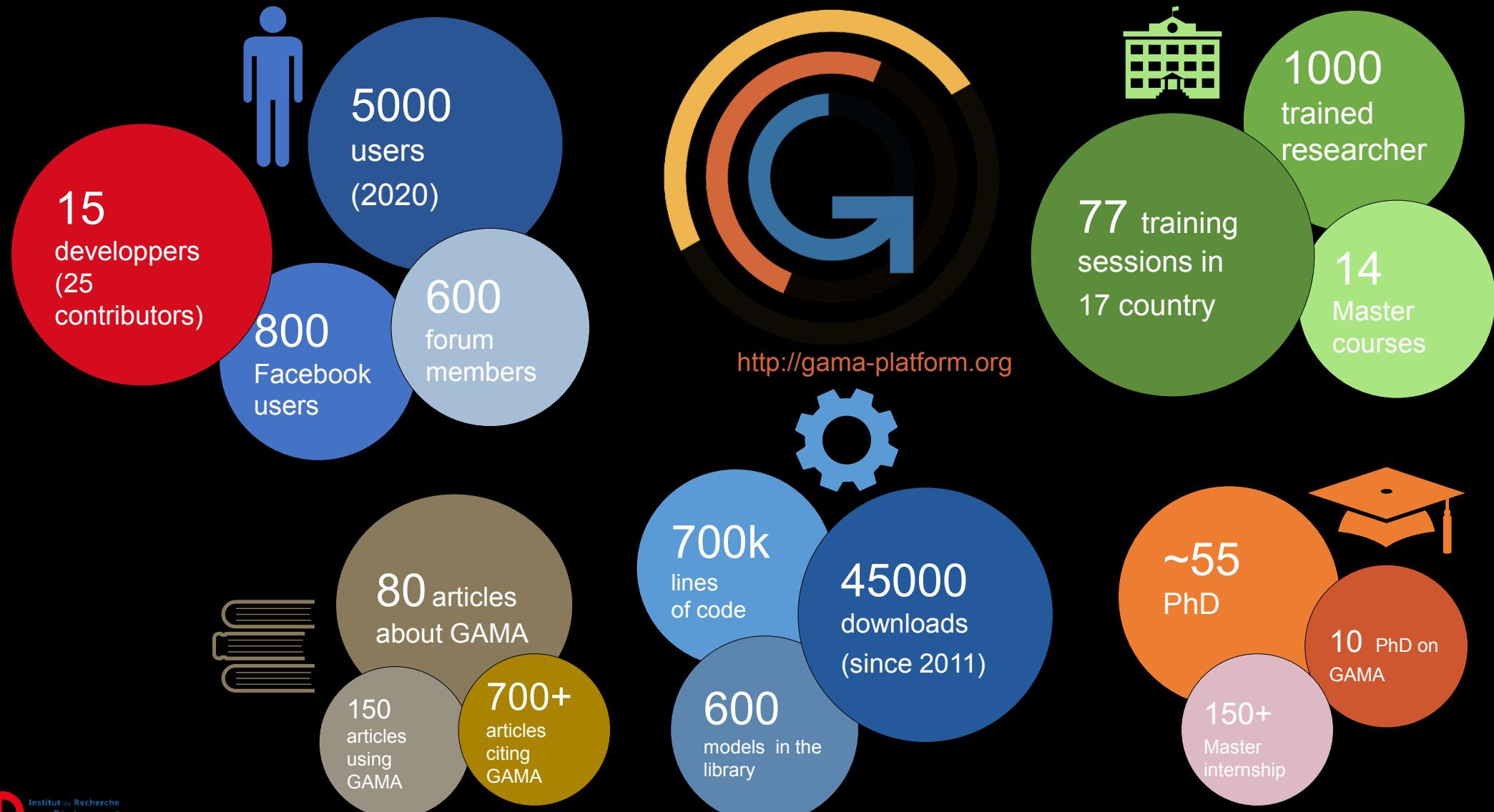


# ACTEUR PROJECT: Building evacuation in crisis context

Valette, M., Gaudou, B., Longin, D., & Taillandier, P. (2018, October). Modeling a Real-Case Situation of Egress Using BDI Agents with Emotions and Social Skills. In *International Conference on Principles and Practice of Multi-Agent Systems*(pp. 3-18). Springer, Cham.



# GAMA IN A FEW STATISTICS...



# More information

- **Websites of the platform**

<http://gama-platform.org>

<https://github.com/gama-platform/gama>

- **Youtube Channel:** gama Modeling

<https://www.youtube.com/channel/ UCWJ1kWGDDI-9u2f2uD0gcaQ/feed>

- **Social Network:** <https://www.facebook.com/GamaPlatform>

- **Mailing-lists**

*General mailing-list:*

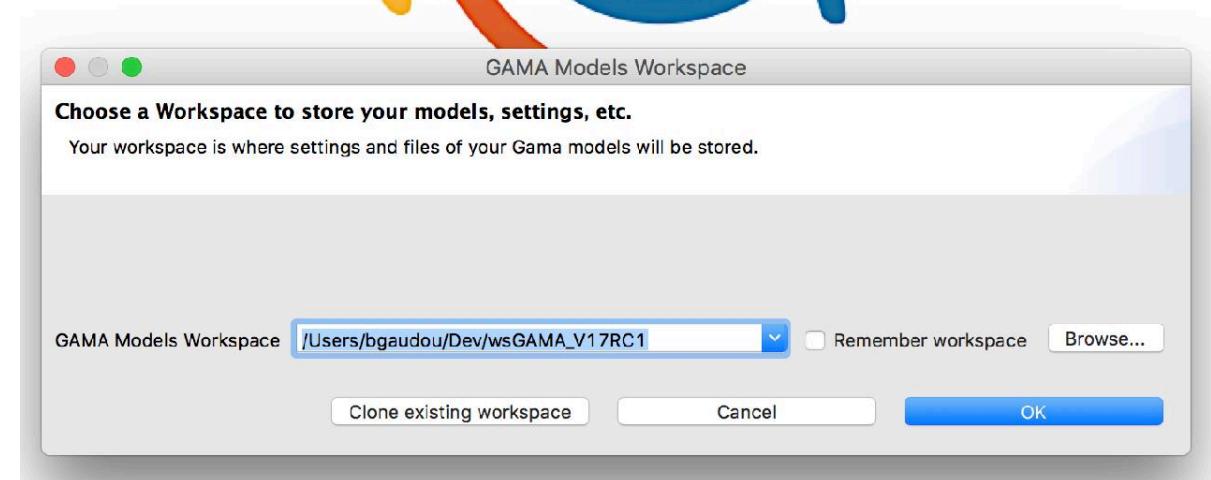
<https://groups.google.com/forum/?fromgroups#!forum/gama-platform>

*Developers mailing-list:*

<https://groups.google.com/forum/?fromgroups#!forum/gama-dev>

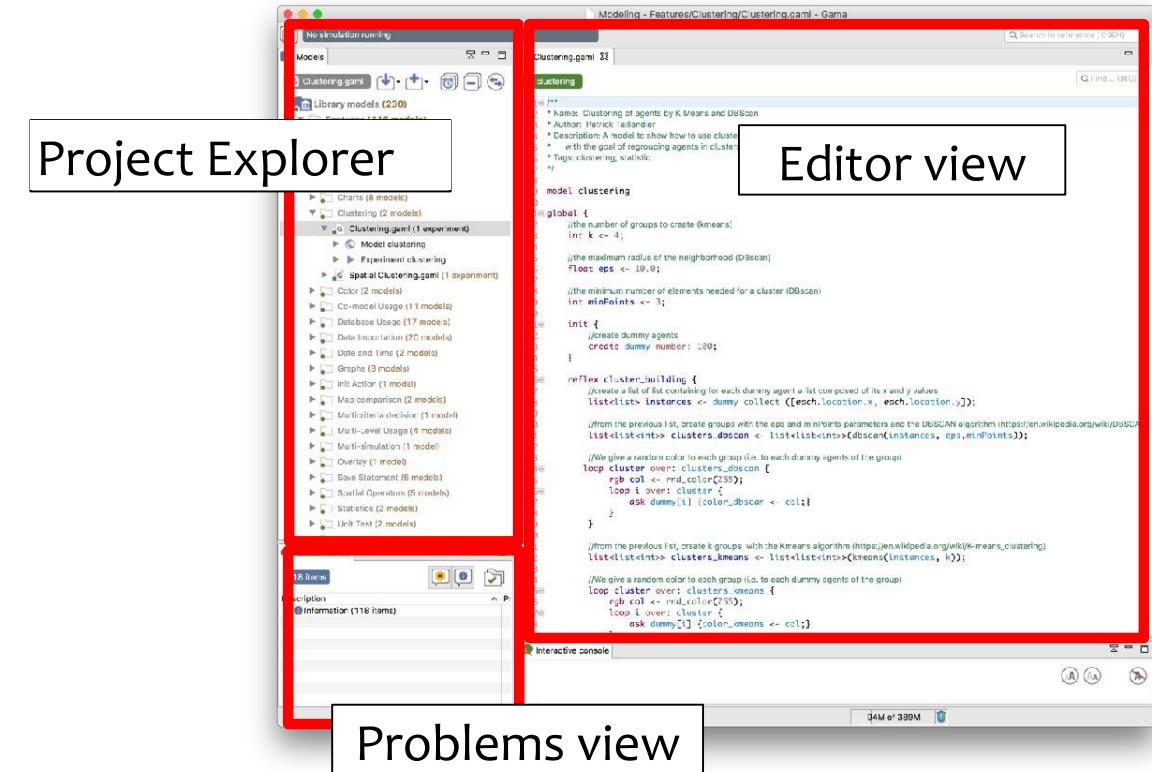
# IT IS NOW TIME TO RUN GAMA !

- First GAMA asks you to choose a workspace.
- A workspace is a folder that will contain all your own projects and models.
- You are free to choose the folder you want!



# GAMA MODEL FILES ARE STORED IN PROJECTS

- Each project may contain several models, as well as additional resources (GIS data, pictures,...).
- Belonging to the same project allows models to use each other and have access to the same resources.
- Projects can be organised in any way, although a default layout (“models”, “doc”, ...) is proposed.



GAMA provides a library of models composed of 7 types of models:

- **Data**: simple models that show how manipulate data (importation, export, database...)
- **GAML Syntax**: models that present the syntax of the GAML language
- **Model Exploration**: models that show how (model calibration, multi-simulation...)
- **Modelling**: models that presents many modeling features of GAMA (spatial component, model coupling...)
- **Toy models**: classic agents-based models (ant models, boids, Schelling...)
- **Tutorials**: models linked to online tutorials
- **Visualization and User interaction**: models that show how to define visualizations and user-interactions

# TAKE A LOOK AT “GAME OF LIFE” MODEL IN LIBRARY

- Models library \ Toy models \ Life \ Life.gaml

The model can be experimented

The model has errors

A screenshot of the Modeler application interface. The title bar says "Modeling - Toy Models/Life". The main window shows a tree view of models under "Library models (230)", including "Features (116 models)" and "Syntax (5 models)". Under "Toy Models (79 models)", there are sub-folders like "Ants (Foraging and Sorting)", "Articles", "Boids", "Bubble Sort", "Circle", and "Clock". On the right, the code editor displays the "Game of Life" model file "Life.gaml". The code is as follows:

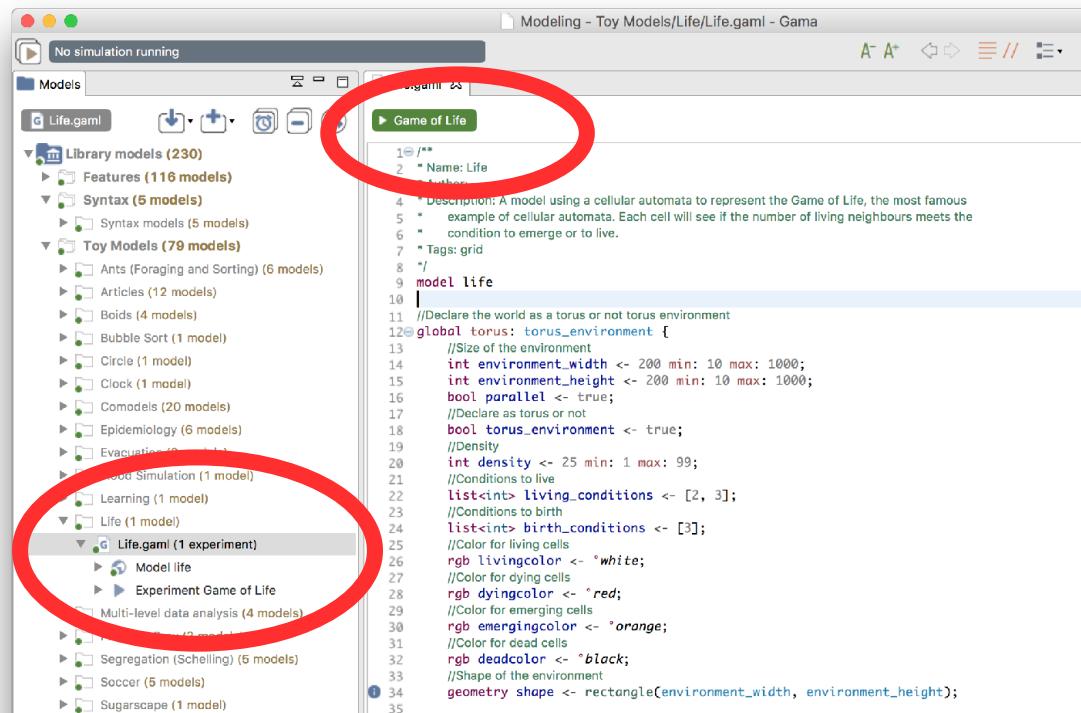
```
1 /**
2 * Name: Life
3 * Author:
4 * Description: A model using a cellular automata to
5 *   example of cellular automata. Each cell will
6 *   condition to emerge or to live.
7 * Tags: grid
8 */
9 model life
10
11 //Declare the world as a torus or not torus environment
12 global torus: torus_environment {
13   //Size of the environment
14   int environment_width <- 200 min;
15   int environment_height <- 200 min;
16 }
```

A screenshot of the Modeler application interface, similar to the previous one but with errors. The title bar says "Modeling - Toy Models/Life". The main window shows the same tree view of models. On the right, the code editor displays the "Game of Life" model file "Life.gaml". The code is identical to the previous screenshot, but an error is highlighted in the status bar at the bottom left: "Error(s) detected". The error message in the status bar is "10 gl".

```
1 /**
2 * Name: Life
3 * Author:
4 * Description: A model using a cellular automata to
5 *   example of cellular automata. Each cell will
6 *   condition to emerge or to live.
7 * Tags: grid
8 */
9 model life
10 gl
11 //Declare the world as a torus or not torus environment
12 global torus: torus_environment {
13   //Size of the environment
14   int environment_width <- 200 min;
15   int environment_height <- 200 min;
16 }
```

# LAUNCH EXPERIMENT GAME OF LIFE

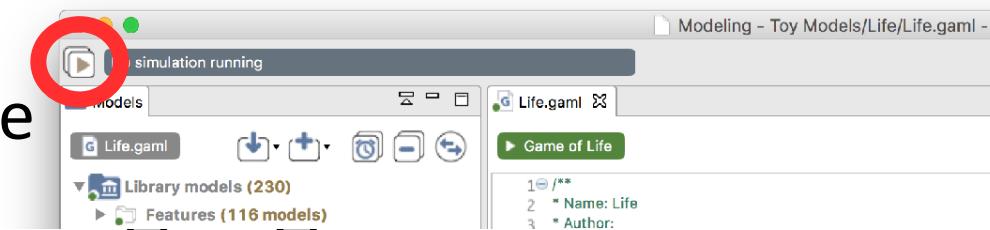
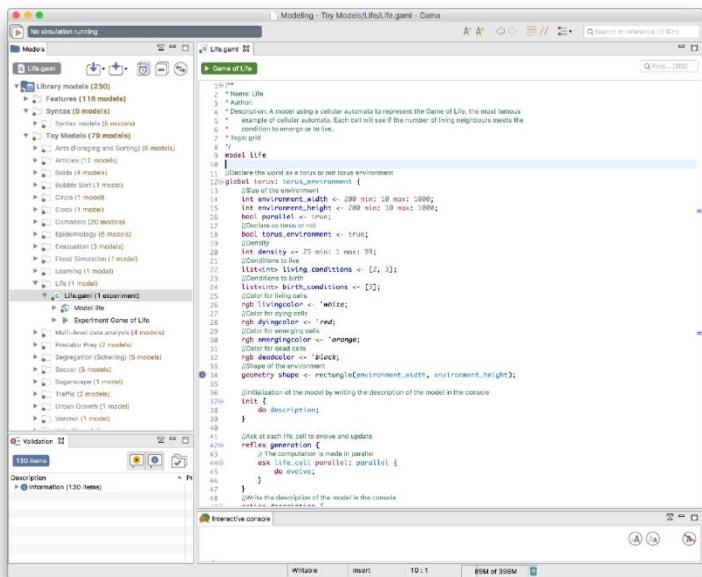
- An experiment is a way to “run” a model. It can be reached either by:
  - ▶ Clicking on an Experiment button
  - ▶ in the Project Explorer, under the name of the model



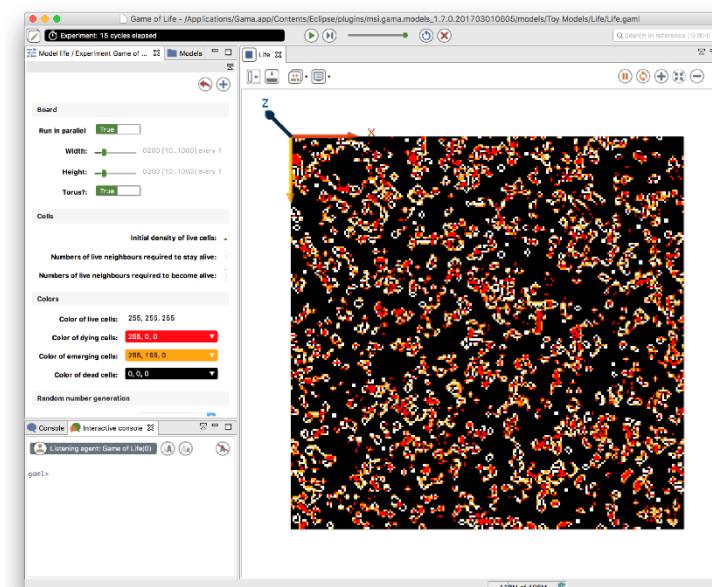
## LAUNCHING AN EXPERIMENT WILL SWITCH FROM *MODELLING* TO *SIMULATION* PERSPECTIVE

- Allows to change perspective

Modelling perspective

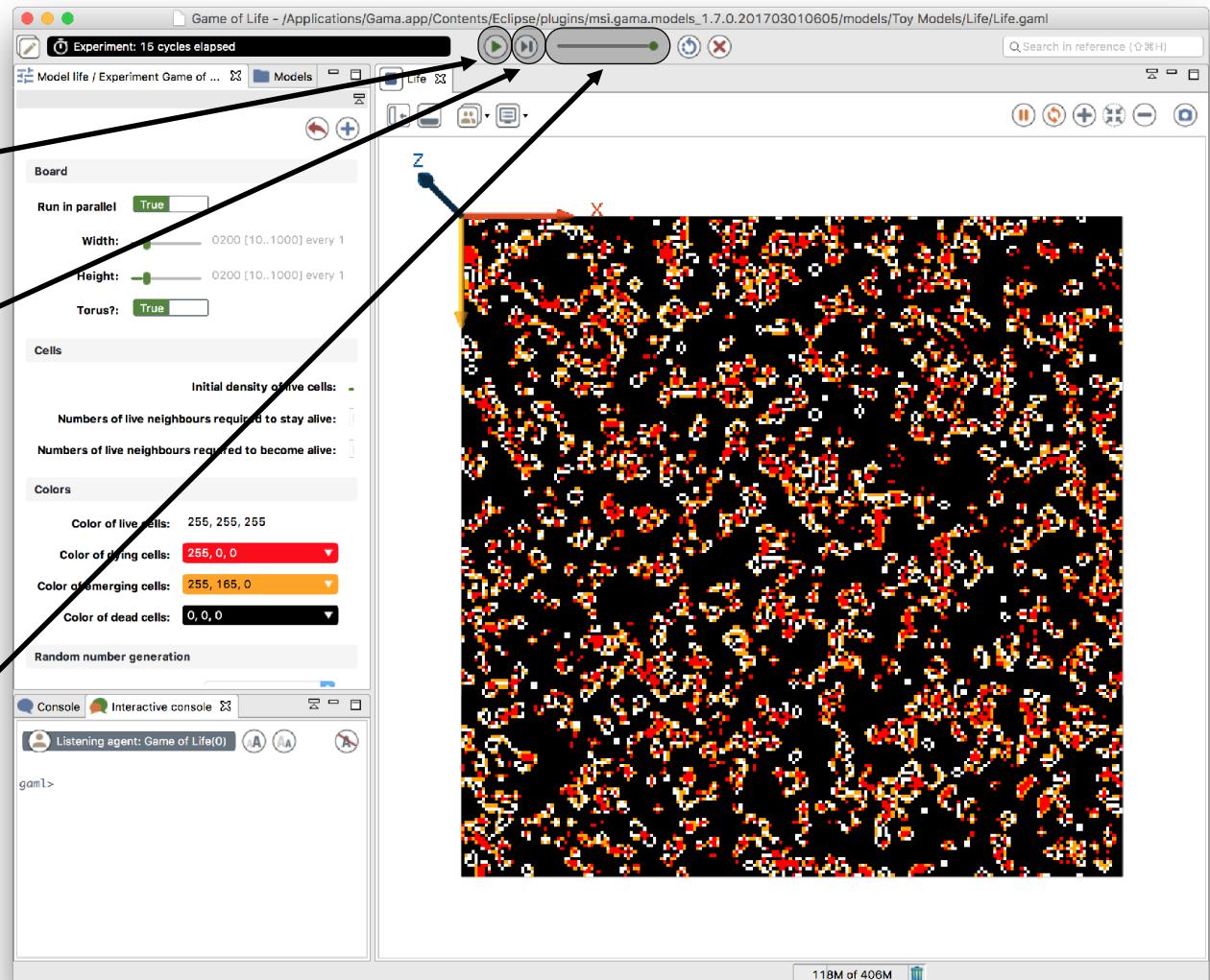


Simulation perspective



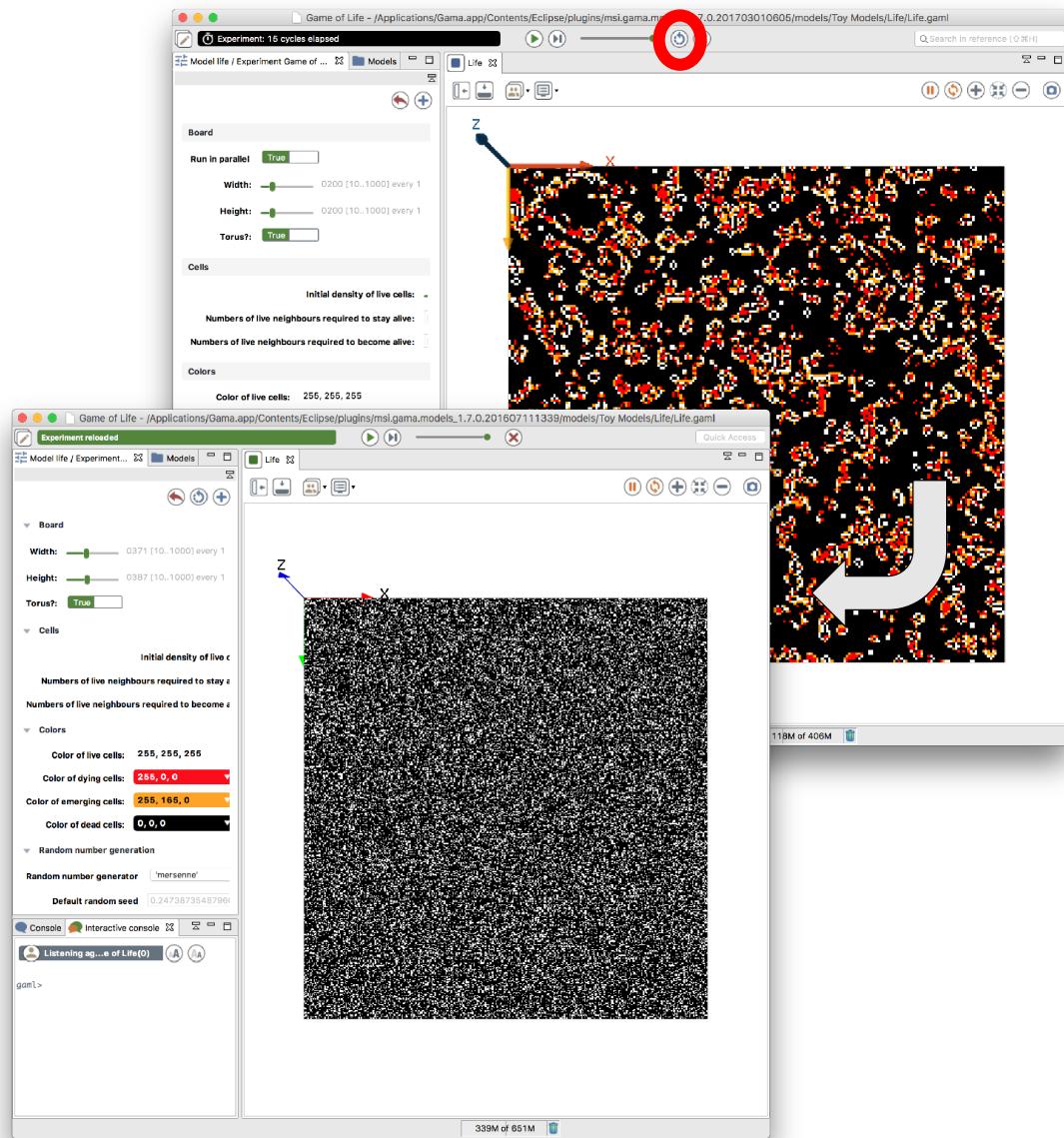
# EXPLORING THE SIMULATION PERSPECTIVE

- Start/pause simulation (it will run until pause is clicked again)
- Step the simulation (it will run one cycle of the simulation)
- Adjust the speed of the simulation

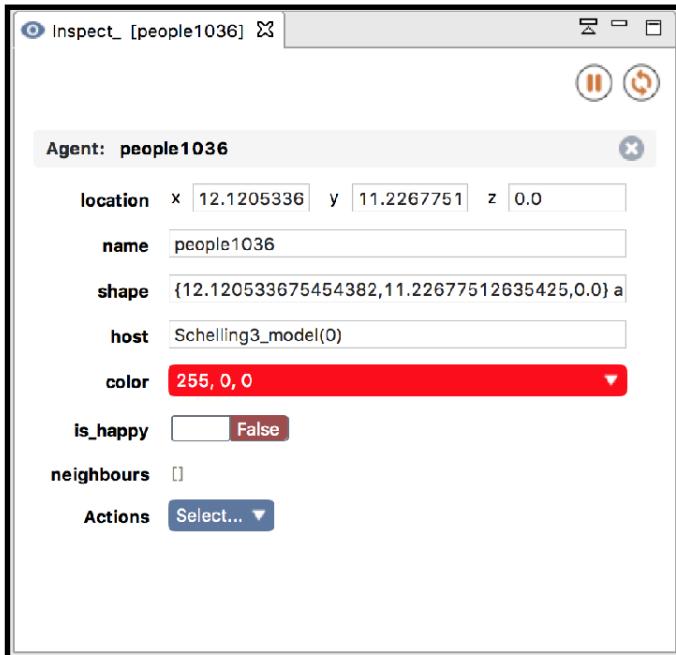


# EXPLORE THE SIMULATION WITH PARAMETERS MODIFIED FROM PARAMETERS VIEW

- The modifications made to the parameters are either:
  - ▶ Used for the current simulation when it makes sense (for instance, if the user changes a color)
  - ▶ Used when the user reloads the experiment otherwise (for instance, if the user changes the size of the grid)
- Launching experiment again (from the model editor) will erase the modifications.



# GAMA OFFERS 2 VIEWS THAT DISPLAY INFORMATION ABOUT ONE OR SEVERAL AGENTS



agent inspector

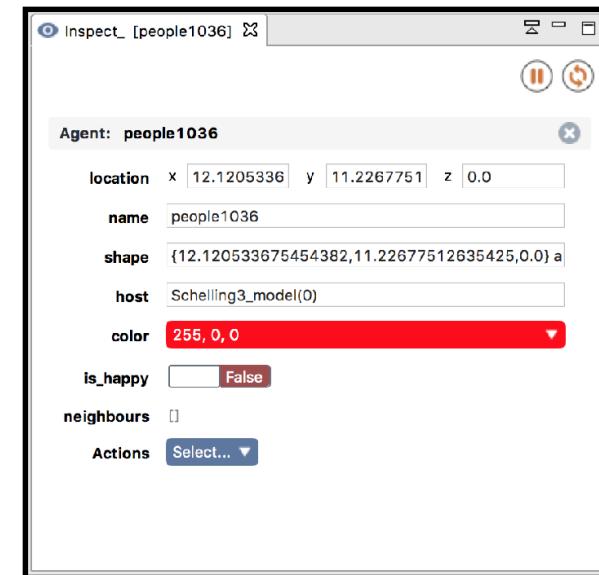
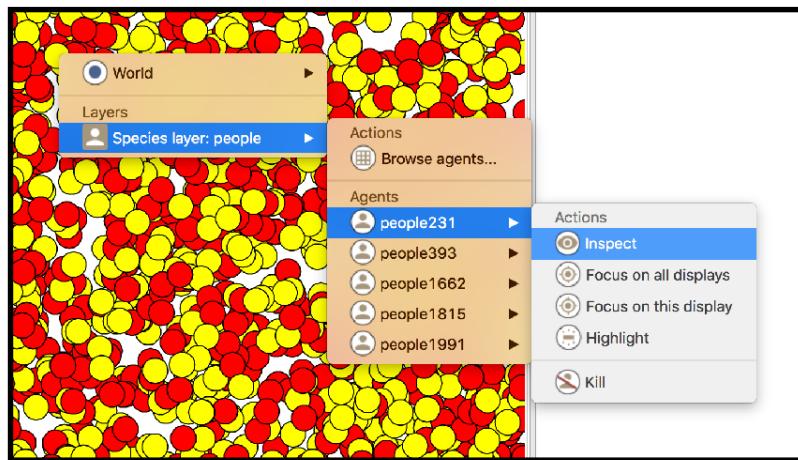
The screenshot shows the 'Browse(0): population of SimpleAgent' window. It lists 20 SimpleAgent entries with the following columns: #, color, location, name, and opinion. The 'Attributes' sidebar on the left shows the available agent properties.

#	color	location	name	opinion
0	rab (103. 57. 13...	{51.454867655...	'SimpleAgent0'	0.4425164033...
1	rab (254. 120. 1...	{37.713065300...	'SimpleAgent1'	0.5687268742...
2	rab (143. 215. 1...	{37.508191731...	'SimpleAgent2'	0.6235161370...
3	rab (42. 202. 10...	{92.256108104...	'SimpleAgent3'	0.5663720528...
4	rab (226. 120. 2...	{97.290306920...	'SimpleAgent4'	0.6658831244...
5	rab (182. 7. 218...	{51.469727905...	'SimpleAgent5'	0.6311607664...
6	rab (25. 117. 11...	{25.560744310...	'SimpleAgent6'	0.7791995483...
7	rab (46. 79. 78...	{75.709297793...	'SimpleAgent7'	0.5887268742...
8	rab (44. 98. 229...	{33.386883396...	'SimpleAgent8'	0.2130192266...
9	rab (167. 78. 18...	{58.936932627...	'SimpleAgent9'	0.5029072021...
10	rab (191. 76. 40...	{7.0288356905...	'SimpleAgent10'	0.5932985490...
11	rab (66. 193. 19...	{49.410029641...	'SimpleAgent11'	0.6982848563...
12	rab (58. 76. 107...	{10.728018127...	'SimpleAgent12'	0.4935022410...
13	rab (138. 98. 31...	{15.423154176...	'SimpleAgent13'	0.6093212645...
14	rab (99. 91. 145...	{20.736089647...	'SimpleAgent14'	0.6311607664...
15	rab (96. 171. 67...	{88.825467574...	'SimpleAgent15'	0.4816172639...
16	rab (180. 87. 70...	{34.349619171...	'SimpleAgent16'	0.4935022410...
17	rab (54. 45. 78...	{39.225633940...	'SimpleAgent17'	0.5932985490...
18	rab (67. 223. 55...	{16.062299931...	'SimpleAgent18'	0.5964384083...
19	rab (189. 93. 24...	{40.014702015...	'SimpleAgent19'	0.6602867719...

agent browser

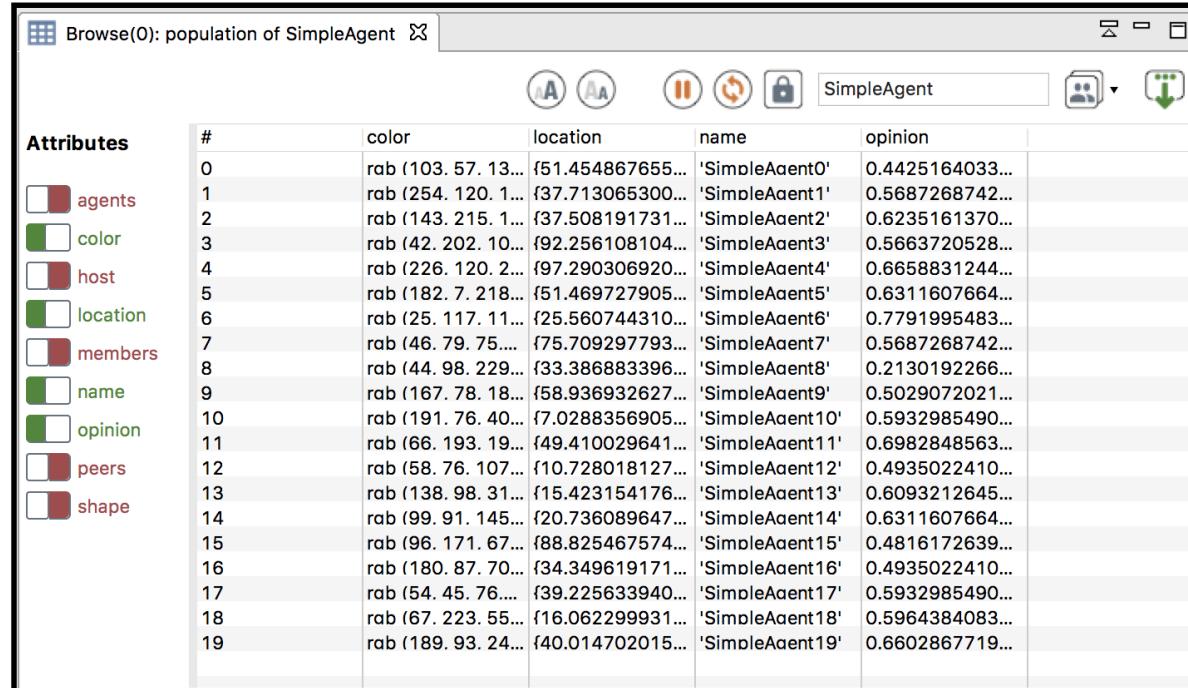
# INSPECT BY RIGHT CLICKING ON A AGENT IN A DISPLAY

- provides information about one specific agent. It also allows to change the values of its variables during the simulation.
- It is possible to «highlight» the selected agent.



# INSPECT INFORMATIONS BY AGENT BROWSER

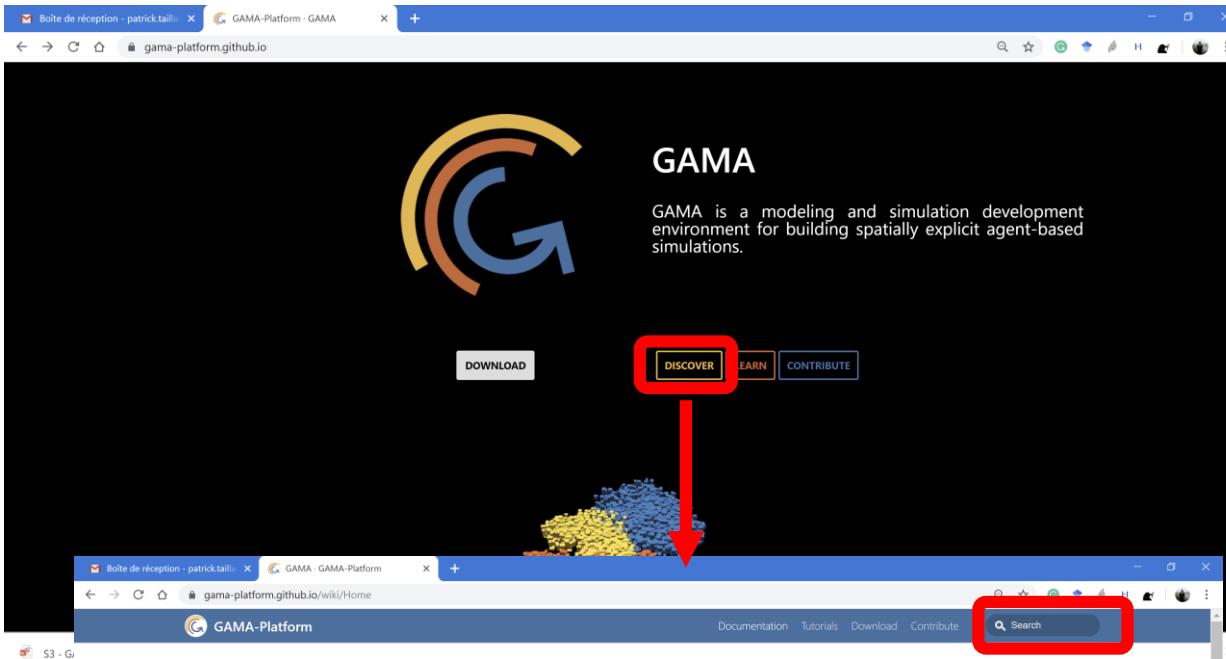
- The species browser provides informations about all or a selection of agents of a species.
- The agent browser is available through the **Agents** menu or by right clicking on a by right\_clicking on a display



The screenshot shows the 'Browse(0): population of SimpleAgent' window. On the left, there is a sidebar titled 'Attributes' with checkboxes for 'agents', 'color', 'host', 'location', 'members', 'name', 'opinion', 'peers', and 'shape'. The main area is a table with columns: '#', 'color', 'location', 'name', and 'opinion'. The table contains 20 rows of data for 'SimpleAgent' objects numbered 0 to 19. The 'color' column lists various coordinates, 'location' lists latitude-longitude pairs, 'name' lists names like 'SimpleAgent0' to 'SimpleAgent19', and 'opinion' lists numerical values between 0.4 and 0.6.

#	color	location	name	opinion
0	rab (103. 57. 13...	f51.454867655...	'SimpleAaent0'	0.4425164033...
1	rab (254. 120. 1...	f37.713065300...	'SimpleAaent1'	0.5687268742...
2	rab (143. 215. 1...	f37.508191731...	'SimpleAaent2'	0.6235161370...
3	rab (42. 202. 10...	f92.256108104...	'SimpleAaent3'	0.5663720528...
4	rab (226. 120. 2...	f97.290306920...	'SimpleAaent4'	0.6658831244...
5	rab (182. 7. 218...	f51.469727905...	'SimpleAaent5'	0.6311607664...
6	rab (25. 117. 11...	f25.560744310...	'SimpleAaent6'	0.7791995483...
7	rab (46. 79. 75...	f75.709297793...	'SimpleAaent7'	0.5687268742...
8	rab (44. 98. 229...	f33.386883396...	'SimpleAaent8'	0.2130192266...
9	rab (167. 78. 18...	f58.936932627...	'SimpleAaent9'	0.5029072021...
10	rab (191. 76. 40...	f7.0288356905...	'SimpleAaent10'	0.5932985490...
11	rab (66. 193. 19...	f49.410029641...	'SimpleAaent11'	0.6982848563...
12	rab (58. 76. 107...	f10.728018127...	'SimpleAaent12'	0.4935022410...
13	rab (138. 98. 31...	f15.423154176...	'SimpleAaent13'	0.6093212645...
14	rab (99. 91. 145...	f20.736089647...	'SimpleAaent14'	0.6311607664...
15	rab (96. 171. 67...	f88.825467574...	'SimpleAaent15'	0.4816172639...
16	rab (180. 87. 70...	f34.349619171...	'SimpleAaent16'	0.4935022410...
17	rab (54. 45. 76...	f39.225633940...	'SimpleAaent17'	0.5932985490...
18	rab (67. 223. 55...	f16.062299931...	'SimpleAaent18'	0.5964384083...
19	rab (189. 93. 24...	f40.014702015...	'SimpleAaent19'	0.6602867719...

# GAMA: HOW TO GET DOCUMENTATION?

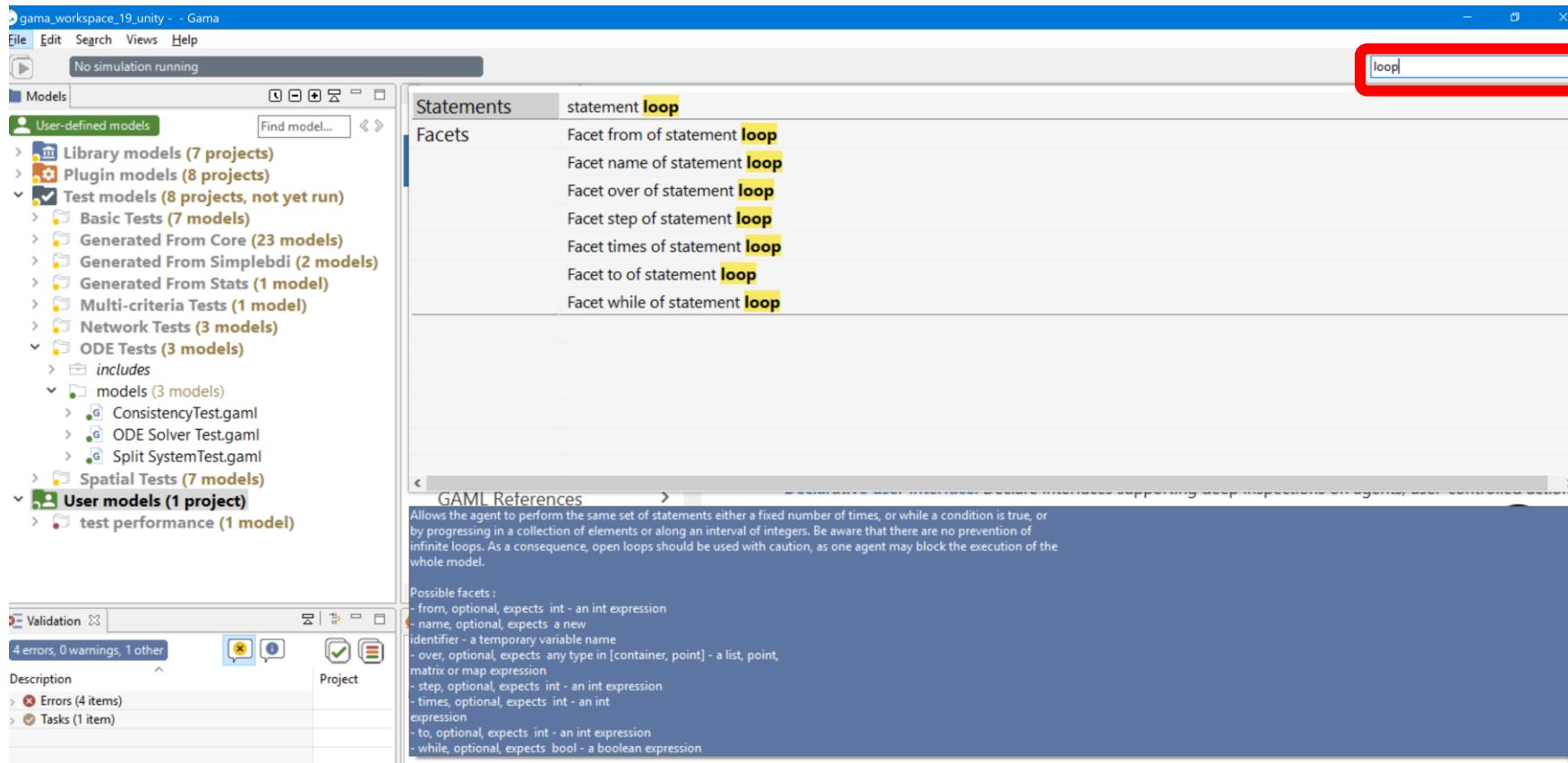


The Website of GAMA proposes a complete documentation of the platform and offers a search engine.

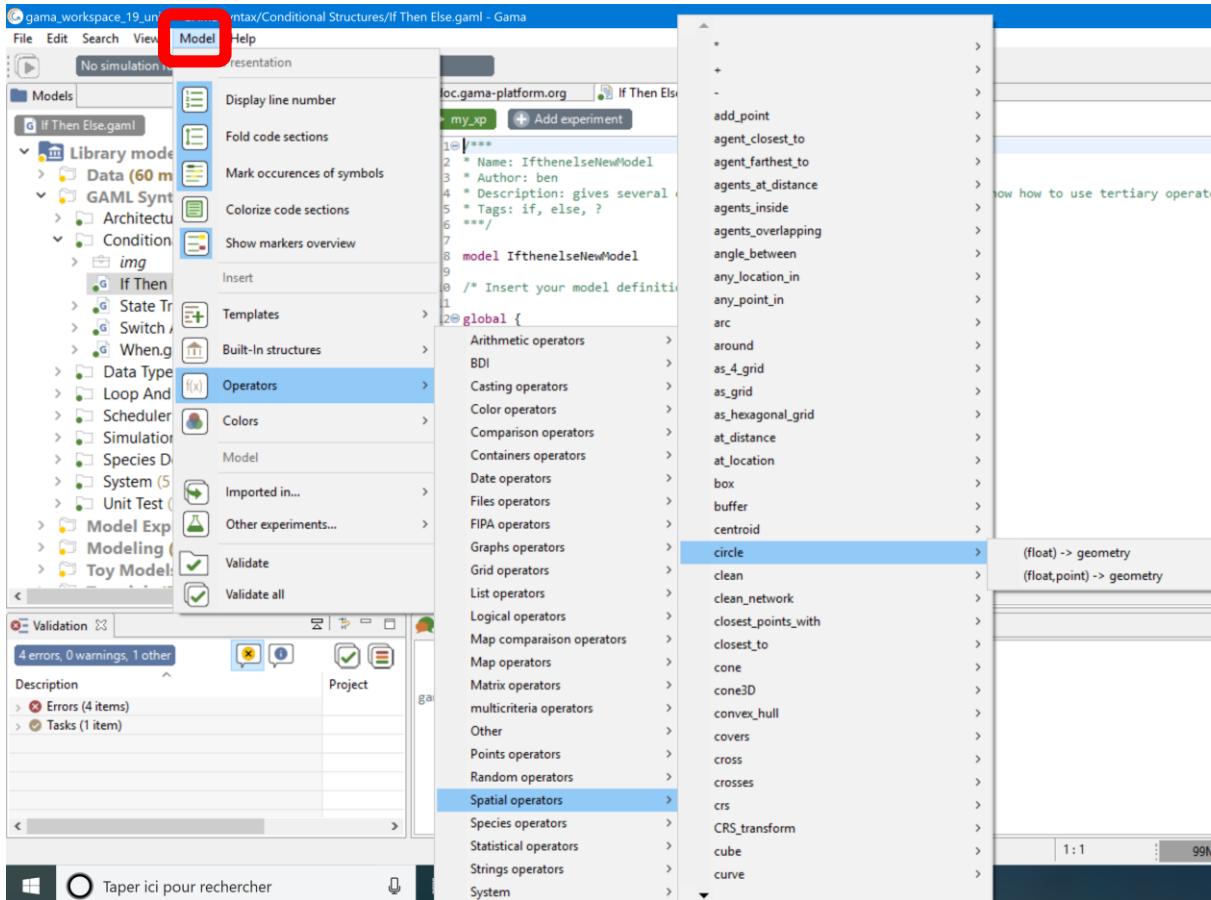
<http://gama-platform.org>

# GAMA: HOW TO GET DOCUMENTATION?

A search engine is also offered by GAMA



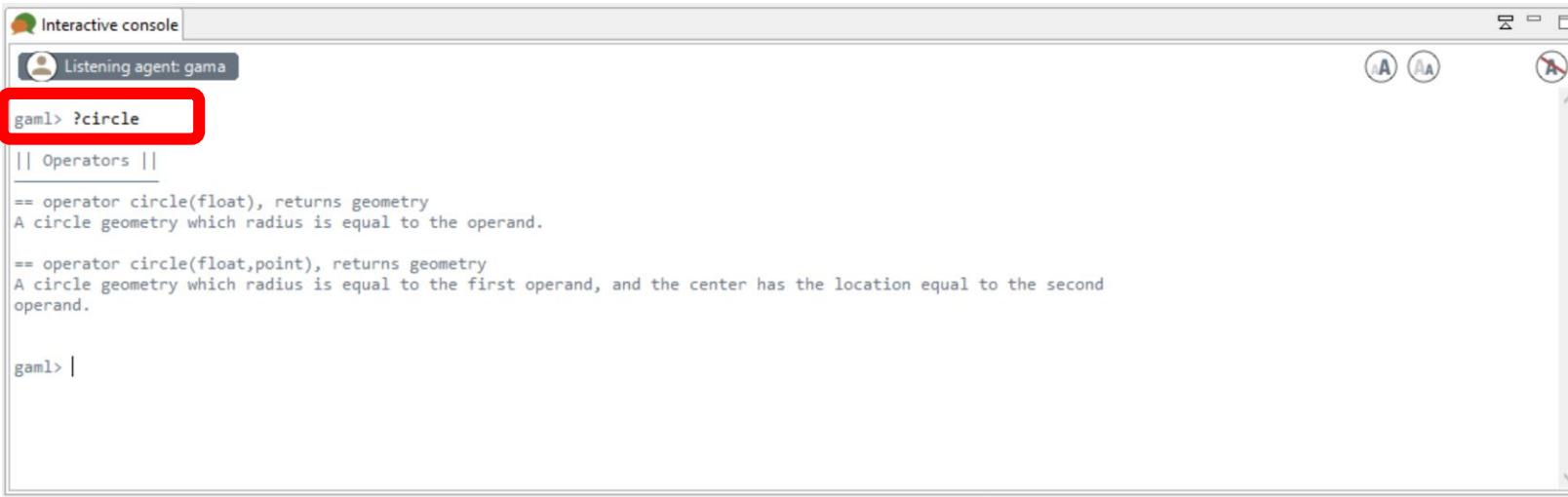
# GAMA: HOW TO GET DOCUMENTATION?



In modeling perspective, a list of all operators and statements is available.

# GAMA: HOW TO GET DOCUMENTATION?

The interactive console enables to interact with a simulation, but also to ask questions to GAMA about a keyword. Just use « ? » before the keyword to have all the documentation about this keyword.



The screenshot shows the GAMA Interactive console window. The title bar says "Interactive console". The status bar at the top left says "Listening agent: gama". The main area contains a command-line interface. A red box highlights the command "gaml> ?circle". Below the command, the output shows documentation for the circle keyword:

```
gaml> ?circle
|| Operators ||
== operator circle(float), returns geometry
A circle geometry which radius is equal to the operand.

== operator circle(float,point), returns geometry
A circle geometry which radius is equal to the first operand, and the center has the location equal to the second
operand.

gaml> |
```

# IMPORT EXISTING PROJECTS INTO THE WORKSPACE

