FLASH-MAS: towards a state-of-the-art, modular agent deployment environment 

Andrei Olaru

[andrei.olaru@upb.ro]

AI-MAS Group
National University of Science and Technology POLITEHNICA Bucharest

03.09.2025

- FLASH-MAS
- Vision
- Structure
- Example
- Deployment
- Visualization
- Monitoring

FLASH-MAS: towards a state-of-the-art, modular agent deployment environment



overview

FLASH-MAS

Easy for beginners, powerful to experts.

[https://github.com/andreiolaru-ro/FLASH-MAS]

Vision

Easy for beginners, powerful to experts.

- multiple applications – network-deployed, local simulation, ABMS
- modular implementations – different implementation for an entity $\not\Rightarrow$ different implementations for other entities
- uniform approach to the implementation of all types of entities
- uniform means of interaction among all entities, via *waves*

Vision

Easy for beginners, powerful to experts.

- multiple applications – network-deployed, local simulation, ABMS
- modular implementations – different implementation for an entity $\not\Rightarrow$ different implementations for other entities
- uniform approach to the implementation of all types of entities
- uniform means of interaction among all entities, via *waves*

Vision

Easy for beginners, powerful to experts.

- multiple applications – network-deployed, local simulation, ABMS
- modular implementations – different implementation for an entity $\not\Rightarrow$ different implementations for other entities
- uniform approach to the implementation of all types of entities
- uniform means of interaction among all entities, via *waves*

Vision

Easy for beginners, powerful to experts.

- multiple applications – network-deployed, local simulation, ABMS
- modular implementations – different implementation for an entity $\not\Rightarrow$ different implementations for other entities
- uniform approach to the implementation of all types of entities
- uniform means of interaction among all entities, via *waves*

Structure

Principles

- all persistent components of the application are *entities*.

Structure

"Standard" entities:

- nodes
- pylons
- agents
- shards

Any other "custom" entities, e.g.:

- drivers
- agent groups
- custom agent executors
- artifacts
- etc

Principles

- all persistent components of the application are *entities*.

Structure

"Standard" entities:

- nodes
- pylons
- agents
- shards

Any other "custom" entities, e.g.:

- drivers
- agent groups
- custom agent executors
- artifacts
- etc

Principles

- all persistent components of the application are *entities*.
- entities interact via *waves*.

Structure

"Standard" entities:

- nodes
- pylons
- agents
- shards

Any other "custom" entities, e.g.:

- drivers
- agent groups
- custom agent executors
- artifacts
- etc

Principles

- all persistent components of the application are *entities*.
- entities interact via *waves*.
- waves are *routed* among entities.

Structure

"Standard" entities:

- nodes
- pylons
- agents
- shards

Any other "custom" entities, e.g.:

- drivers
- agent groups
- custom agent executors
- artifacts
- etc

Principles

- all persistent components of the application are *entities*.
- entities interact via *waves*.
- waves are *routed* among entities.
- entities can be placed in the *context* of other entities.

Demo Example

An agent receives images – e.g. from the autonomous driving domain – and applies people detection on those images and sends the images to other agents, depending on the number of people in the image.

Demo Example

An agent receives images – e.g. from the autonomous driving domain – and applies people detection on those images and sends the images to other agents, depending on the number of people in the image.

I want to

- deploy the MAS
- follow the progress of the MAS – e.g. how many images have been processed so far
- see which images are received by each agent

Deployment

```
-load_order driver;pylon;agent
-package eumas testing -loader agent:composite
-node node1 central:web
  -pylon local:pylon1
    -agent Sorter -shard gui from:sorter-gui.yaml -shard PeopleSorter script:people
    -agent NoPeople -shard Receptor -shard gui from:receptor-gui.yaml
    -agent FewPeople -shard Receptor -shard gui from:receptor-gui.yaml
    -agent MorePeople -shard Receptor -shard gui from:receptor-gui.yaml
```

with all agents also having `-shard messaging -shard remoteOperation`

```
-load_order driver;pylon;agent
-package eumas testing -loader agent:composite
-node node1 central:web
  -driver ML:mldriver                              << custom 'driver' entity type
    -pylon local:pylon1
      -agent Sorter -shard gui from:sorter-gui.yaml -shard PeopleSorter script:people
      -agent NoPeople -shard Receptor -shard gui from:receptor-gui.yaml
      -agent FewPeople -shard Receptor -shard gui from:receptor-gui.yaml
      -agent MorePeople -shard Receptor -shard gui from:receptor-gui.yaml
```

- shard gets a reference to the `MLDriver`

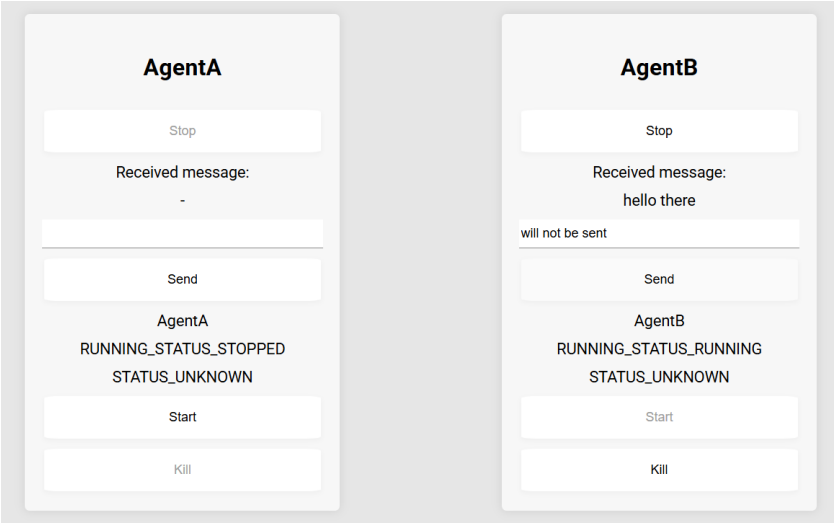
  ```
  if(context instanceof MLDriver) mlDriver = (MLDriver) context;
  ```

- call functionality of the `MLDriver`

  ```
  mlDriver.predict("YOLOv8-pedestrians", filename);
  ```

Visualization

```
- type:  label
  port:  inbound
  role:
recvd-label
  value:  "Received
message:"
- type:  label
  port:  inbound
  role:  content
  value:  "-"
- type:  form
  port:  outbound
  role:  content
- type:  button
  port:  outbound
  role:  send
  value:  Send
```



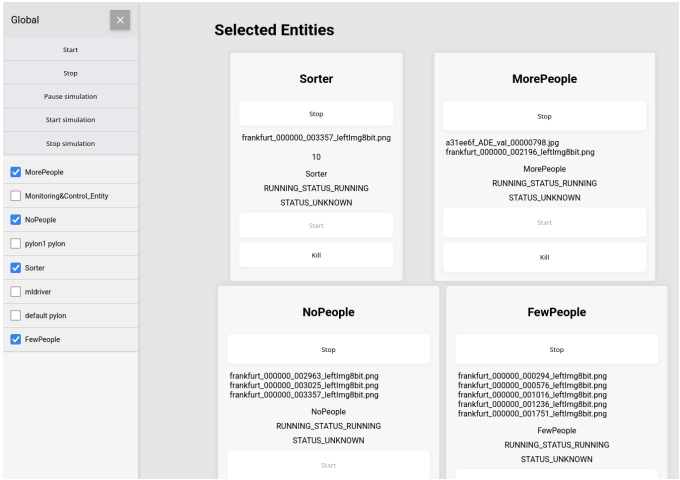
Visualization

Sorter:

```
- port:  current-image
  value:  "no image"
- port:  nprocessed
  value:  0
```

Receivers:

```
- port:  received
  value:  "none"
```



Monitoring

```
[FSM] Detected event 'Prediction result' from log line. Transitioning...
```

```
---------------------------- Log lines processed: 382
6 predictions done:
 > [ PeopleSorter               ] Prediction result: [[[5.0, 55525.293296575546]]] / [5.0]
 > [ PeopleSorter               ] Prediction result: [[[1.0, 2406.40541991964]]] / [1.0]
 > [ PeopleSorter               ] Prediction result: [[[1.0, 5795.841445313767]]] / [1.0]
 > [ PeopleSorter               ] Prediction result: [[[3.0, 38927.3651953144]]] / [3.0]
 > [ PeopleSorter               ] Prediction result: [[[2.0, 6318.080986328423]]] / [2.0]
 > [ PeopleSorter               ] Prediction result: [[[2.0, 7331.842226562556]]] / [2.0]

 1 5-people predictions
 7 warnings
 0 errors


 ---------------------------- ^
```

```
> [ PeopleSorter               ] Getting prediction for [frankfurt_000000_002196_leftImg8bit.png]
                                > [ testing-Sorter >>>>          ] Sending a message from [Sorter/remote] to [Monitoring&Control_Entity/ENTITY_GUI_OUTPUT/current-image] with content [frankfurt_000000_002196_leftImg8bit.png].
                                > [ testing-Sorter >>>>          ] Sending a message from [Sorter/remote] to [Monitoring&Control_Entity/ENTITY_GUI_OUTPUT/nprocessed] with content [7].
                                . [ M&C                          ] Routing wave[{EVENT_TYPE=[AGENT_WAVE], content=[frankfurt_000000_002196_leftImg8bit.png], destination-element=[ENTITY_GUI_OUTPUT, current-image], source-element=[Sorter, remote],
                                > [ Monitoring&Control_Entity.gui ] Sending output wave: [{EVENT_TYPE=[AGENT_WAVE], content=[frankfurt_000000_002196_leftImg8bit.png], destination-element=[Sorter, current-image], source-element=[Sorter, remote], de
                                . [ M&C                          ] Routing wave[{EVENT_TYPE=[AGENT_WAVE], content=[7], destination-element=[ENTITY_GUI_OUTPUT, nprocessed], source-element=[Sorter, remote], destination-complete=[Monitoring&Control_
                                > [ Monitoring&Control_Entity.gui ] Sending output wave: [{EVENT_TYPE=[AGENT_WAVE], content=[7], destination-element=[Sorter, nprocessed], source-element=[Sorter, remote], destination-complete=[Sorter/nprocessed]}]
[FSM] Detected event 'Getting prediction' from log line. Transitioning...
<ML server>  returned <Response 39 bytes [200 OK]>

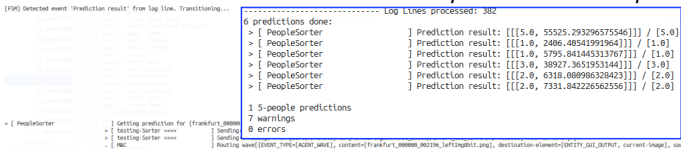
image 1/1 /home/andreiolaru/Andrei/Dev/Workspace/Flash-MAS/ml-directory/input/frankfurt_000000_002196_leftImg8bit.png: 320x640 6 persons, 7 cars, 1 bus, 1 truck, 2 traffic lights, 116.6ms
Speed: 1.6ms preprocess, 116.6ms inference, 29.5ms postprocess per image at shape (1, 3, 320, 640)
127.0.0.1 - - [03/Sep/2025 00:11:57] "POST /predict HTTP/1.1" 200 -
                                > [ mldriver                     ] Response: [{"prediction":[[6,29168.658632813254]]}]
                                > [ mldriver                     ] Prediction: [[[6.0, 29168.658632813254]]]
> [ PeopleSorter               ] Prediction result: [[[6.0, 29168.658632813254]]] / [6.0]
                                > [ testing-Sorter >>>>          ] Sending a message from [Sorter] to [MorePeople] with content [frankfurt_000000_002196_leftImg8bit.png].
                                > [ MorePeople.Receptor          ] received content:[frankfurt_000000_002196_leftImg8bit.png]
> [ testing-MorePeople >>>>    ] Sending a message from [MorePeople/remote] to [Monitoring&Control_Entity/ENTITY_GUI_OUTPUT/received] with content [a31ee6f_ADE_val_00000798.jpg
frankfurt_000000_002196_leftImg8bit.png].
                                . [ MorePeople.remote            ] Agent [MorePeople] event unhandled [{EVENT_TYPE=[AGENT_WAVE], content=[frankfurt_000000_002196_leftImg8bit.png], destination-complete=[MorePeople], source-element=[Sorter]}].
                                . [ M&C                          ] Routing wave[{EVENT_TYPE=[AGENT_WAVE], content=[a31ee6f_ADE_val_00000798.jpg
frankfurt_000000_002196_leftImg8bit.png], destination-element=[ENTITY_GUI_OUTPUT, received], source-element=[MorePeople, remote], destination-complete=[Monitoring&Control_Entity/ENTITY_GUI_OUTPUT/received]}]
> [ testing-MorePeople >>>>    ] agent [MorePeople] event: [{EVENT_TYPE=[AGENT_WAVE], content=[frankfurt_000000_002196_leftImg8bit.png], destination-complete=[MorePeople], source-element=[Sorter]}]
                                > [ Monitoring&Control_Entity.gui ] Sending output wave: [{EVENT_TYPE=[AGENT_WAVE], content=[a31ee6f_ADE_val_00000798.jpg
frankfurt_000000_002196_leftImg8bit.png], destination-element=[MorePeople, received], source-element=[MorePeople, remote], destination-complete=[MorePeople/received]}]
[FSM] Detected event 'Prediction result' from log line. Transitioning...
[FSM] Detected event 'AGENT_WAVE' from log line. Transitioning...
                    7 predictions done: [ . . . . . . . . ] 1 5-people predictions 7 warnings 0 errors
```

Log summarization (2) State verification

Monitoring

```
- e:  "PeopleSorter"
  level:  ">"
  match:  ["Prediction result"]
  out:  ["count", "predictions
done:","list:full"]
  expect:  ["count", "40"]
- e:  "PeopleSorter"
  level:  ">"
  match:  ["Prediction result",
"5.0"]
  out:  ["count", "5-people
predictions"]
  expect:  ["count", "10"]
- e:  "*"
  level:  "*"
  out:  ["count", "warnings"]
  expect:  ["count", "7"]
- e:  "*"
  level:  "#"
  out:  ["count", "errors"]
  expect:  ["count", "0"]
```

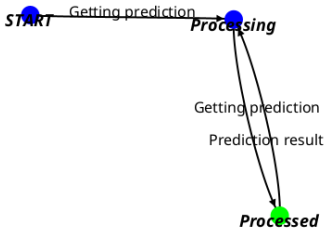
periodic block report



permanent terminal footer

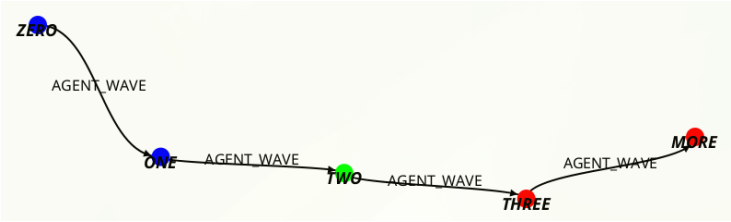
```
- e:  "PeopleSorter"
  level:  ">"
  fsm-from:  "sorter.dot"
  fsm-start:  "START"
  fsm-expect:  "Processed"
```

```
digraph Automaton {
  START -> Processing [ label="Getting prediction" ];
  Processing -> Processed [ label="Prediction result" ];
  Processed -> Processing [ label="Getting prediction" ]; }
```



```
- e:  "MorePeople"
  level:  ">"
  fsm-from:  "morepeople.dot"
  fsm-start:  "ZERO"
  fsm-expect:  "MORE"
```

```
digraph Automaton {
  ZERO -> ONE [ label="AGENT_WAVE" ];
    ONE -> TWO [ label="AGENT_WAVE" ];
      TWO -> THREE [ label="AGENT_WAVE" ];
        THREE -> MORE [ label="AGENT_WAVE" ]; }
```



Thank You!

Questions are welcome!



This work was supported by the European Union's HORIZON Research and Innovation Programme under grant agreement No 101120657, project ENFIELD (European Lighthouse to Manifest Trustworthy and Green AI).

[andrei.olaru@upb.ro]