The entity-operation model for practical multi-entity deployment

Andrei Olaru, Gabriel Nicolae and Adina Magda Florea

andrei.olaru@upb.ro

AI-MAS Group, University Politehnica of Bucharest

31.05.2023

- Context

- Model

- Implementation

- Conclusion

The entity-operation model for practical multi-entity deployment

overview

Problem Context

What should things be modeled as?

Problem Context

What should things be modeled as?

How easy is it for a developer to translate the model into an implementation?

Problem Context

In a smart building, there are various rooms containing smart devices. When Alice is entering a room, and when her device connects to the local access point, the context manager in the room contacts Alice's device and the devices in the room become available to her. Her device also receives notifications about events related to the room. The context manager detects when Alice leaves the room.

Problem Context

In a smart building, there are various rooms containing smart devices. When Alice is entering a room, and when her device connects to the local access point, the context manager in the room contacts Alice's device and the devices in the room become available to her. Her device also receives notifications about events related to the room. The context manager detects when Alice leaves the room.

How should the context manager be modeled? Is it agent or artifact?

How should the context manager contact Alice's device?

Are they events or are they messages?

Problem Context

To control the blinds, an agent needs to send a command to a Raspberry Pi, via a system of ROS nodes.

Problem Context

To control the blinds, an agent needs to send a command to a Raspberry Pi, via a system of ROS nodes.

Who should be the recipient of the command?

Is it the blinds controller or the ROS gateway?

What should the command be represented as?

Is it a message or something else?

Problem Context

In the AI Folk project, agents form a "culture" in which they are able to search for, transfer, and exchange machine learning (ML) models and use them depending on the situation. ML models can be queried for results, can be transferred like objects, and migrate in the system together with their owner agents.

Problem Context

In the AI Folk project, agents form a "culture" in which they are able to search for, transfer, and exchange machine learning (ML) models and use them depending on the situation. ML models can be queried for results, can be transferred like objects, and migrate in the system together with their owner agents.

How should an ML model be modeled as?

Is it an artifact, an agent, or just a different kind of object?

Problem Context

Artifacts / elements of environment – receive operations to perform.

Agents (reactive) – receive perceptions and perform / respond with actions.

Agents (cognitive) – receive messages [and perceptions] and perform actions.

Resource / space / activity managers – receive notifications regarding included entities.

Broadcast groups – receive messages and broadcast to the group.

Support / communication infrastructures – route interactions.

Nodes – manage entity execution.

Sub-agent entities – provide functionality.

Problem Context

Artifacts / elements of environment – receive operations to perform from authorized agents / from agents in the workspace.

Agents (reactive) – receive perceptions and perform / respond with actions.

Agents (cognitive) – receive messages [and perceptions] and perform actions.

Resource / space / activity managers – receive notifications regarding included entities.

Broadcast groups – receive messages and broadcast to the group.

Support / communication infrastructures – route interactions.

Nodes – manage entity execution.

Sub-agent entities – provide functionality.

Problem Context

Artifacts / elements of environment – receive operations to perform from authorized agents / from agents in the workspace.

Agents (reactive) – receive perceptions from the environment and perform / respond with actions.

Agents (cognitive) – receive messages [and perceptions] and perform actions.

Resource / space / activity managers – receive notifications regarding included entities.

Broadcast groups – receive messages and broadcast to the group.

Support / communication infrastructures – route interactions.

Nodes – manage entity execution.

Sub-agent entities – provide functionality.

Problem Context

Artifacts / elements of environment – receive operations to perform from authorized agents / from agents in the workspace.

Agents (reactive) – receive perceptions from the environment and perform / respond with actions.

Agents (cognitive) – receive messages [and perceptions] and perform actions.

Resource / space / activity managers – receive notifications from authorized entities regarding included entities.

Broadcast groups – receive messages and broadcast to the group.

Support / communication infrastructures – route interactions.

Nodes – manage entity execution.

Sub-agent entities – provide functionality.

Problem Context

Artifacts / elements of environment – receive operations to perform from authorized agents / from agents in the workspace.

Agents (reactive) – receive perceptions from the environment and perform / respond with actions.

Agents (cognitive) – receive messages [and perceptions] and perform actions.

Resource / space / activity managers – receive notifications from authorized entities regarding included entities.

Broadcast groups – receive messages from group members and broadcast them.

Support / communication infrastructures – route interactions.

Nodes – manage entity execution.

Sub-agent entities – provide functionality.

Problem Context

Artifacts / elements of environment – receive operations to perform from authorized agents / from agents in the workspace.

Agents (reactive) – receive perceptions from the environment and perform / respond with actions.

Agents (cognitive) – receive messages [and perceptions] and perform actions.

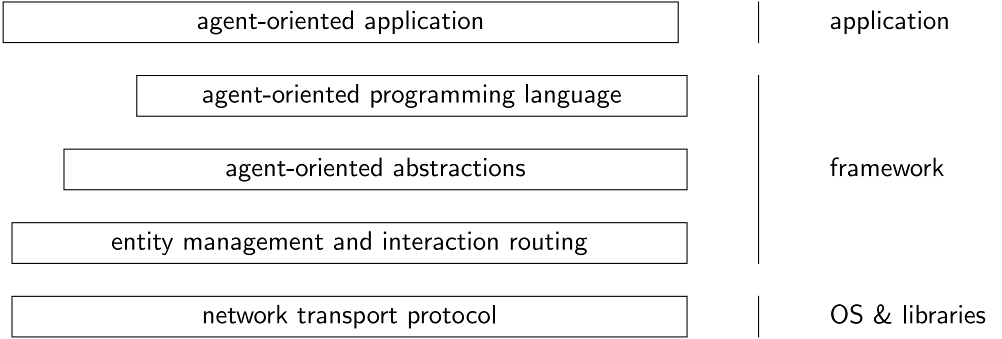
Resource / space / activity managers – receive notifications from authorized entities regarding included entities.

Broadcast groups – receive messages from group members and broadcast them.

Support / communication infrastructures – route interactions.

Nodes – manage entity execution.

Sub-agent entities – provide functionality to owner agents.



agent-oriented application

agent-oriented programming language

agent-oriented abstractions

entity management and interaction routing

network transport protocol

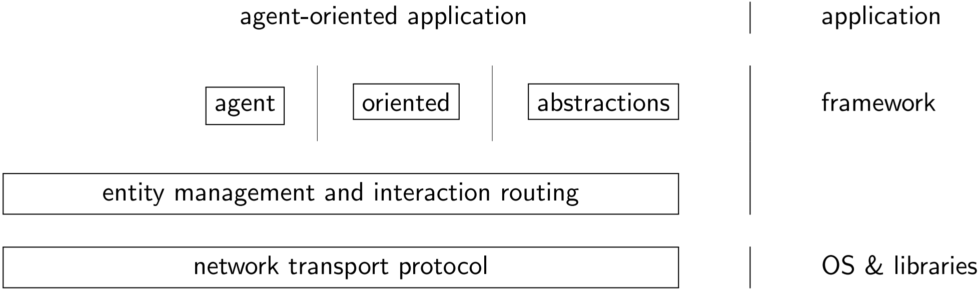
application

framework

OS & libraries

abstraction – e.g. an agent, an artifact, a node, etc.

Entity-Operation Model



agent-oriented application

agent

oriented

abstractions

entity management and interaction routing

network transport protocol

application

framework

OS & libraries

abstraction – e.g. an agent, an artifact, a node, etc.

- the manner of interaction differs greatly depending on the abstraction
- abstractions have very different implementations and manners of access
- difficult to access abstractions for framework infrastructure (e.g. communication services)

Entity-Operation Model

We introduce a uniform means of interacting with entities in a MAS.

This brings improved interoperation and openness, without affecting existing models for MAS, plus the ability to model more types of entities.

Entity-Operation Model

- Entity – any element in the multi-agent system that is *persistent* for some time and has a certain level of autonomy.

  For example: agents, artifacts, nodes, service infrastructures

  Entities can be

  - *local* to a node (e.g. agents)
  - *distributed* over several nodes (e.g. groups, workspaces) and *embodied* on at least one node

Entity-Operation Model

- Entity – any element in the multi-agent system that is persistent for some time and has a certain level of autonomy.

  For example: agents, artifacts, nodes, service infrastructures

  Entities can be

  - *local* to a node (e.g. agents)
  - *distributed* over several nodes (e.g. groups, workspaces) and *embodied* on at least one node

- Operation of an entity – a means for other entities to interact with that entity.
  - operations have *arguments*
  - operations *may* have *return values*

Entity-Operation Model

Access to operations *may* be restricted, based on the relations of the caller with other entities.

For instance, the *broadcast* operation of a *broadcast group* may be called only by an entity which is a *member* of the group.

Entity-Operation Model

Access to operations *may* be restricted, based on the relations of the caller with other entities.

For instance, the *broadcast* operation of a *broadcast group* may be called only by an entity which is a *member* of the group.

An operation call contains the arguments for the call and the context tokens that prove the caller has the relations required for the operation.

Context tokens are issued by an entity the callee trusts (or by the callee itself).

For every member, a *broadcast group* issues a context tokens proving that $m \xrightarrow{\textit{part-of}} \textit{group}$.

For an entity E in a given room R, a location service issues a context token proving that $E \xrightarrow{in} R$, which can be used by a context manager offering services to E.

Entity-Operation Model

A MAS is formed of entities and relations: $\langle EE, RR \rangle$, with

the entities: $EE = \{E \mid E = \langle ID_E, Ops_E \rangle\}$ and

relations: $RR = \{\langle from, relation, to \rangle\}, \text{with } from, to \in EE$

An operation $O \in Ops_E$: $O = \langle Name_O, Description_O, Args_O, Result_O, Restrictions_O \rangle$

Restrictions: $Restrictions_O \subseteq \{Conjunction \mid Conjunction \subseteq \mathcal{R} \times EE\}$, with
$\mathcal{R} = \{relation \mid \langle *, relation, * \rangle \in RR\}$

An operation call: $\langle E_{Source}, E_{Destination}, Name_{Op}, \{Arguments\}, \{Tokens\}, send\text{-}result \rangle$

Entity-Operation Model

What is an agent?

In JADE, an agent is an entity which can *receive* a message from another agent.

Entity-Operation Model

What is an agent?

In JADE, an agent is an entity which can *receive* a message from another agent.

or

In JaCaMo, an agent is an entity which can *receive* a message from another agent or *perceive* a change in the environment (in an artifact).

Entity-Operation Model

What is an agent?

In JADE, an agent is an entity which can *receive* a message from another agent.

or

In JaCaMo, an agent is an entity which can *receive* a message from another agent or *perceive* a change in the environment (in an artifact).

or

A reactive is an agent which can *perceive* the environment and respond with an action.

Entity-Operation Model

What is an agent?

In JADE, an agent is an entity which can *receive* a message from another agent.

or

In JaCaMo, an agent is an entity which can *receive* a message from another agent or *perceive* a change in the environment (in an artifact).

or

A reactive is an agent which can *perceive* the environment and respond with an action.

or

In ABMS an agent is an entity which *receive* events and *perceive* the environment.

Entity-Operation Model

An artifact receives hiliteAaction / observe / focus requests, from entities in the correct workspace.

A workspace is a distributed entity which can receive join and leave requests.

A support infrastructure receives route requests.

A node can receive start or stop requests from authorized entities (e.g. entity owners), and can welcome mobile agents from other nodes.

A shard receives requests for specific functionality from its owner agent.

Any entity supports the list operation.

Implementation challenges

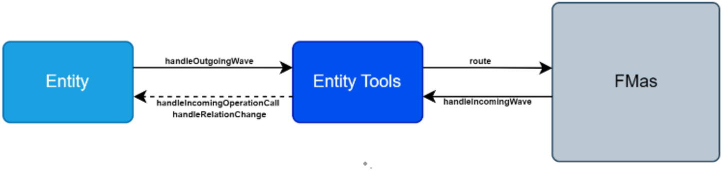
We implemented the entity-operation model in the FLASH-MAS framework, with several highlights: [https://github.com/andreiolaru-ro/FLASH-MAS]

- the framework (FMAS) as a "soup" in which all entities live, a thin layer which handles operation calls.

Implementation challenges

We implemented the entity-operation model in the FLASH-MAS framework, with several highlights: [https://github.com/andreiolaru-ro/FLASH-MAS]

- the framework (`FMAS`) as a "soup" in which all entities live, a thin layer which handles operation calls.
- each entity is given an `EntityTools` instance to interact with `FMAS` and to manage operation calls.



Implementation challenges

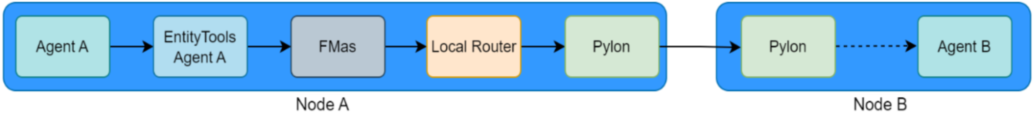
We implemented the entity-operation model in the FLASH-MAS framework, with several highlights: [https://github.com/andreiolaru-ro/FLASH-MAS]

- the framework (`FMAS`) as a "soup" in which all entities live, a thin layer which handles operation calls.
- each entity is given an `EntityTools` instance to interact with `FMAS` and to manage operation calls.
- there are several types of "waves" to route between entities – operation calls, operation results, relation initiations, relation acknowledgement.

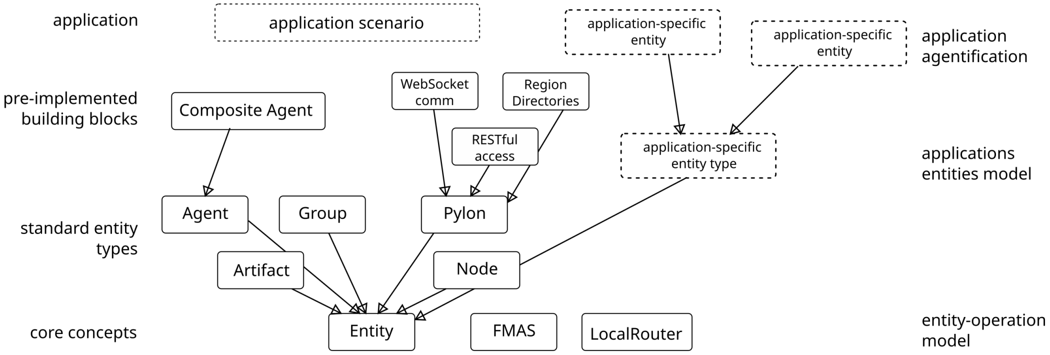
Implementation challenges

We implemented the entity-operation model in the FLASH-MAS framework, with several highlights: [https://github.com/andreiolaru-ro/FLASH-MAS]

- the framework (FMAS) as a "soup" in which all entities live, a thin layer which handles operation calls.
- each entity is given an EntityTools instance to interact with FMAS and to manage operation calls.
- there are several types of "waves" to route between entities – operation calls, operation results, relation initiations, relation acknowledgement.
- there must be a Local Router to route waves between different communication infrastructures.



Implementation challenges



Implementation challenges

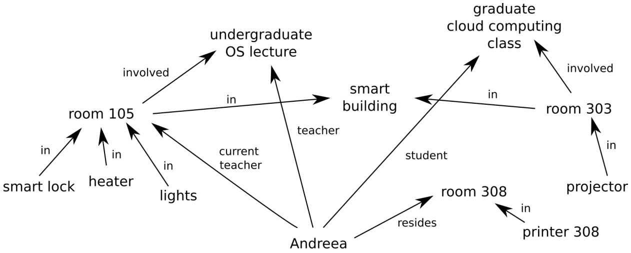
We have implemented an Ambient Intelligence scenario using entities which rely on the entity-operation model to:

- access operations depending on their context
- create and remove relations between entities dynamically, both for user roles and for physical location
- implement various types of entities, among which agents, artifacts, context managers, and infrastructure elements.

Implementation challenges

We have implemented an Ambient Intelligence scenario using entities which rely on the entity-operation model to:

- access operations depending on their context
- create and remove relations between entities dynamically, both for user roles and for physical location
- implement various types of entities, among which agents, artifacts, context managers, and infrastructure elements.



Conclusion and future work

- We can describe entities in a MAS in a uniform manner
- We can interoperate entities using a uniform interface
- We can keep existing models but implement them using a uniform model

Conclusion and future work

- We can describe entities in a MAS in a uniform manner
- We can interoperate entities using a uniform interface
- We can keep existing models but implement them using a uniform model

- Rigorously define existing models in terms of entities and operations
- Use the entity-operation model to interoperate with other frameworks
- Develop complex scenarios with dynamic infrastructure topologies
- Model operations that an entity can call

Thank You!

Questions are welcome!

andrei.olaru@upb.ro