Contextualize Agent Interactions by Combining Communication and Physical Dimensions in the Environment

Stéphane Galland[1], Flavien Balbo[2], Nicolas Gaud[1], Sebastian Rodriguez[3], Gauthier Picard[2], and Olivier Boissier[2]

[1] IRTES Institute, Université de Technologie de Belfort-Montbéliard, Belfort, France
firstname.lastname@utbm.fr
[2] Ecole Nationale Supérieure des Mines, 158 Cours Fauriel, 42100 Saint-Etienne, France
firstname.lastname@mines-stetienne.fr
[3] GITIA Laboratory, Facultad Regional Tucumán, Universidad Tecnológica Nacional, San Miguel de Tucumán, Argentina
sebastian.rodriguez@gitia.org

Abstract. The environment, as a space shared between agents, is a key component of multiagent systems (MAS). Depending on systems, this space may integrate physical, communication or communication dimensions. Each of them has its own process and rules to support agents' interaction. The dimensions of the environment are generally connected either outside of the agents or within each agent, according to the target application. In order to ensure a multiagent control, the relations between dimensions must be explicit outside of the agents. Using these relations between the environment dimensions, the interaction becomes also multi-dimensional. In this paper, rules and mechanisms to make this connection outside of the agents are formalized. The model connects the physical and communication dimensions to realize contextualized interactions. It is implemented using the SARL multiagent programming language, and illustrated with an urban traffic simulation.

Keywords: Environment modeling; Simulation; Programming languages for agents and multi-agent systems; Smart cities

1 Introduction

The environment, as a space shared between agents, is a key component of multiagent systems [20]. Depending on systems, this space may integrate physical, communication or social dimensions where agents interact. Each dimension of the environment has its own process and rules to support its interaction model. For instance, in the physical dimension, the rules may be based on the location of the agents, i.e. the interaction between agents is allowed according to the distance or any existing obstacle between them. These rules are independent of the agents, i.e. even if it is initiated by the agents, the interaction occurs independently of the decision process of the agents, and is performed by the environment. When the environment has several dimensions, the issue is

to model and to manage the relations between them. In most systems, the dimensions are considered either independent and connected only through the decision process of the agents. Interactions resulting from the combinations of input information depend of the agent's decision process. The problem is that an agent should not be the place where interaction rules are triggered because the interaction could not rely on its own responsibility. For instance, a rule could be designed to regulate the communication in the social dimension according to information coming from the physical dimension (is the receiver agent physically able to receive the message?). The agent cannot decide by itself. This rule should be triggered by the environment using information coming from the physical and communication dimensions.

This multidimensional point of view on the environment opens new perspectives in the design of contextualized interactions. Interactions could be designed using information coming from different dimensions and interactions between agents are not only the result of the action of an agent in one dimension of the environment but also the potential propagation of the interaction through the other dimensions. For instance, a communication act (an agent sends a message to another) could be influenced by physic conditions. In this case, it may have two effects: an exchange of messages in the communication dimension of the environment, and the propagation of sound in the physical environment. The issue is related to the support in a single model the interaction inside a dimension and the relation between the dimensions. In this paper, a model for combining the communication and physical dimensions of the environment is proposed in order to contextualize interaction between agents. This model is implemented with the SARL[4] agent-oriented programming language that enables to define the different dimensions of the environment based on the same concepts.

This paper is organized as follows. Section 2 describes the services related to the environment and how the interaction is ported inside the dimensions of the environment. The illustrative traffic example is also introduced. Section 3 details the multidimensional model of the environment. Section 4 presents the SARL language, and the implementation key elements of our environment model. Section 5 presents the extension of the model for traffic management and simulation. Section 6 discusses our proposal according to related works. Section 7 concludes and gives perspectives.

2 Dimensions of the Environment

Several researches defined the environment and its role in the design of a multiagent system. Our work is based on the E4MAS[5] (Environment for multi-agent systems) definition of the environment: *The environment is a first-class abstraction that provides the surrounding conditions for agents to exist and that mediates both the interaction among agents and the access to resources.*

Since this definition of the environment being not restrictive, heterogeneous implementations have been proposed. Real or simulated systems are based on a "physical environment" [1,5], where agents and objects have an explicit location and proceed actions that are located too. In these systems, interaction results from these actions.

[4] `http://www.sarl.io`
[5] `https://distrinet.cs.kuleuven.be/events/e4mas/`

Other systems are based on a “communication environment” [17,9], where agents have a social knowledge about the others, and they interact following different modalities (direct, indirect or awareness). In this paper, the “physical environment” and “communication environment” are considered as two observable and accessible dimensions of the environment. Each of them structures the MAS according to specific models and/or implementations.

An issue is to ensure the management of all the dimensions, and their relations, in a normalized way in order to enable new contextualized interactions. The context is defined as *”any information that can be used to characterize the situation of an entity“* [3]. The design of the interaction based on the analysis of information coming from the different dimensions of the environment is at the heart of this paper: the articulation between the physical and communication dimensions for supporting contextualized interactions.

In order to illustrate our proposal and provide a *proof-of-concept*, a multiagent traffic simulation, where vehicles are modeled as agents, is implemented (Figure 1). This application aims to reduce the traffic jams by enabling the vehicles to perceive earlier the traffic state through the communication dimension prior the physical dimension, and adapt their driving behaviors. Each vehicle agent communicates with the others and the infrastructure following a cooperative model (V2X communication). Our example focuses on the following scenario: a vehicle requests the priority at junctions. The agent-agent and agent-infrastructure interactions are discussed.

The physical environment is a set of edges, with sensors giving the traffic density, and junctions. Some of these junctions are equipped with Road Side Units (RSU) controlling traffic signals and relaying messages from/to the Internet from/to the vehi-

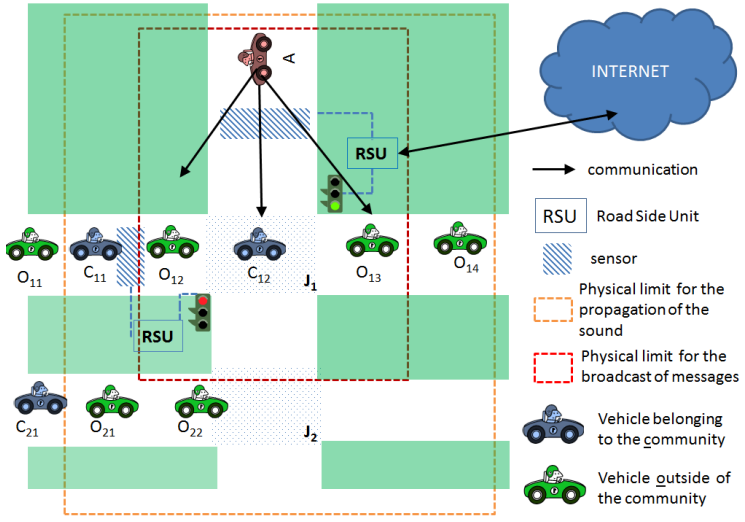

Fig. 1. Traffic simulation example

cles that are equipped with On Board Unit (OBU). The communication environment is composed by vehicles belonging to a community (noted C_x) and vehicles outside of the community (noted O_x). In this community, only emergency vehicles are allowed to request priority by messages and are equipped with a siren, whose sound is propagated following physical laws. The traffic regulation process is based on the dynamic computation of signal plans [2]. An RSU determines the axis with the priority based on the traffic density and existing priority requests. When an emergency vehicle requests the priority, it turns on its siren and sends a priority request at a regular interval. When an RSU receives a priority request, it modifies the traffic plan in order to give a green phase to the vehicle. Its request message, which is completed by an advice, is forwarded by the Internet to all vehicles belonging to the community. There are two junctions in the example. However, only the first, noted J_1, is regulated by an RSU (the second is noted J_2). In the simulation scenario, every vehicle agent slows down when it perceives the sound of a siren without any other information by communication. The agents in the community follow the advice given by the community. Only the vehicle A can request priority.

This simulation scenario illustrates the relations between the dimensions of the environment for interaction in addition of the agent-dimension — or agent-environment — interaction. In the scenario, three cases are considered. The **case a** illustrates when *the interaction in one dimension is constrained by the second*: Coming from the communication environment, the broadcast of the priority request is limited in the physical environment by the V2X propagation model and is extended in the communication environment by the Internet. The **case b** illustrates when *the same interaction has different forms in the two dimensions*: The interaction resulting from the need of the vehicle A has two potential forms: the siren and the message. The propagation of them and their consequences are distinct in each dimension of the environment. The **case c** illustrates when *an interaction initiated by an agent in a dimension generates an interaction in the other dimension*: The priority request in the community is taken into account by the RSU at junction J_1 to adapt dynamically the traffic signal plan in the physical dimension.

3 Environment Model Combining Dimensions

The environment contains elements that are related to its dimensions (Figure 2). Each of these dimensions is associated to a specific model that defines the structure and the dynamics of the environment elements in the dimension.

The physical dimension of the environment contains objects, including the agents' bodies [19]. It can be decomposed into areas, sub-area set, and so on, which are connected together thanks to neighborhood links. The communication dimension may take different forms, as blackboard, sugar space, organizational models, etc. In our example, the communication dimension contains the representations of the community members. The underlying model provides the tools for exchanging messages through the Internet between the OBUs, and the OBUs and the RSUs.

It is basically assumed that a dynamic environment can change of state beyond the agents' control. Introducing the concept of dimension implies each dimension belongs

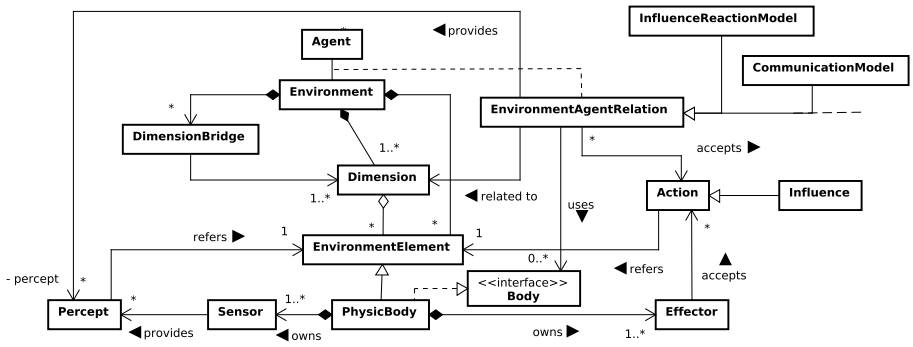

Fig. 2. Environment model combining dimensions, and its relation with the agents

its own dynamic process since it is based on the structure and the rules of the related dimension. For example, the RSU entities are part of the endogenous process of the physical dimension. They receive the priority requests and adapt dynamically the traffic signal plan according to the physical environment rules. In addition to the dynamic processes associated with the dimensions, the global behavior of the environment is considered for managing the interaction between the different dimensions.

Each dimension provides an interaction model for defining how agents are able to interact within this dimension. According to the classical definition of the agent, this model should permit the agent to perceive — provides `Percept` — and act — accept `Action` — in the environment dimension. The concrete definition of the interaction model depends on the dimension's model. A suitable concept to manage the interface between the agents and the physical environment is the agent *body*, *i.e.* a component that is attached to each agent for accessing to one dimension of the environment. A body has a collection of sensors and effectors since they are related to the intrinsic nature and structure of the environment. These sensors and effectors contain a collection of filtering mechanisms that permit to restrict the information that is provided to and received from the agents, respectively. In the example, each agent belonging to the community has two bodies: the vehicle for the physical dimension, and the avatar of the connected device on the Internet for the communication dimension.

For supporting joint actions of the agents in a dimension of the environment, the concept of `Action` is specialized to `Influence` according to the Influence-Reaction model [8]. An influence describes a desired change of the state of an environment dimension. The influences are gathered by the dimension's model. Conflicts among them are detected and solved. And finally, the dimension model is reacting to the influences by applying the resulting state change.

4 Implementation with the SARL language

SARL is a general-purpose agent-oriented programming language [15]. This language aims at providing the fundamental abstractions for dealing with concurrency, distribu-

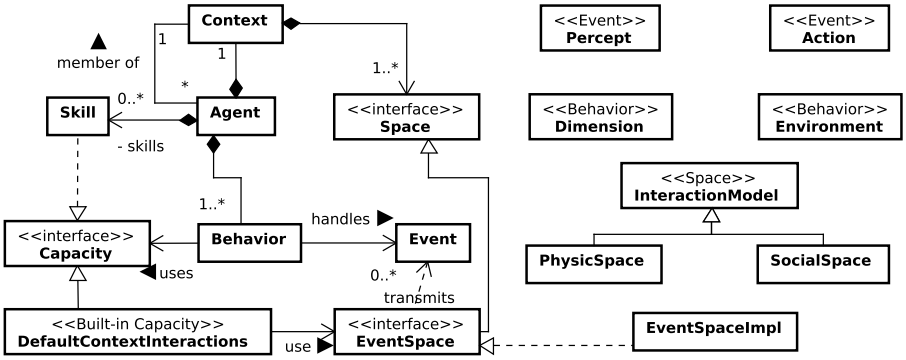

Fig. 3. Major concepts in the SARL metamodel

tion, interaction, decentralization, reactivity, autonomy and dynamic reconfiguration. SARL provides a reduced set of key concepts that are considered as essential for implementing multi-agent systems: Agent, Space, Capacity and Skill.

Space is the abstraction to define *an interaction space between agents or between agents and their environment*. In the SARL toolkit, a concrete default space, which propagates events, called `EventSpace` (and its implementation `EventSpaceImpl`), is proposed. *An **Agent** is an autonomous entity having a set of skills to realize the capacities it exhibits*. An agent has a set of built-in capacities considered essential to respect the commonly accepted competences of agents, like the autonomy, reactivity, pro-activity and social capacities. The agent has the capacity to incorporate behaviors that will determine its global conduct. *Behavior maps a collection of perceptions represented by Events to a sequence of Actions.* By default, the behaviors of an agent communicate using an event-driven approach. *An Event is the specification of some occurrence in a Space that may potentially trigger effects by a listener. A **Capacity** is the specification of a collection of actions*. This specification makes no assumptions about its implementation. It could be used to specify what an agent can do, what a behavior requires for its execution. Indeed, an action is a specification of a transformation of a part of the designed system or its environment. This transformation guarantees resulting properties if the system before the transformation satisfies a set of constraints. *A **Skill** is a possible implementation of a capacity fulfilling all the constraints of this specification*. Each of these generic concepts of the SARL's metamodel is associated to language statements. These statements are used for implementing the multidimensional environment model.

In order to interact in the physical dimension, agents must have a dedicated capacity. The script 1.1 describes the features that are accessible to an agent for all physical environments (`AbstractPhysicEnvironmentCapacity`) and more specifically for the environments related to traffic domain (`RoadEnvironmentCapacity`).

```
1 capacity AbstractPhysicEnvironmentCapacity {
2   def getLinearSpeed: double
3   def setPhysicalPerceptionAlterator(filter: PhysicalPerceptionAlterator)
4   def influence(inf: Influence)
```

```
5   |   def killBody
6   | }
7   | capacity RoadEnvironmentCapacity extends AbstractPhysicEnvironmentCapacity {
8   |   def getPosition: Pair<Edge,double>
9   |   def getOrientation: Direction
10  | }
```

Script 1.1. Interaction capacity with physical Environment

The perception mechanism of the agents in the physical environment depends on the agent's body which contains a geometric description of the perception field. The agent can modify this description in order to alter the physical properties of its sensors (`PhysicalPerceptionAlterator`). When the set of perceptible objects has been computed, the agent receives it in a `Perception` event.

```
1   | event Influence { var object: EnvironmentElement }
2   | event MotionInfluence extends Influence {
3   |   var linearSpline: double
4   |   var linearShift: double
5   |   var path: Edge[]
6   | }
7   | event Siren extends Influence
8   | event Perception { var objects: Percept[] }
```

Script 1.2. Events related to the physical environment

A space dedicated to the physical environment (`PhysicSpace` and `PhysicSpaceImpl`) allows creating an agent's body, to put it in the physical environment and to notify the corresponding model when there are influences related to the body (Script 1.3). For each described capacity, a concrete skill must be defined. This skill (`RoadPhysicSpace`) is used to bind the agent to its body in the physical dimension, and to serve as a gateway between the capacity functions and the space functions (Script 1.3).

```
1   | space PhysicSpace {
2   |   def getBodyFactory: PhysicBodyFactory
3   |   def putInEnvironment(body: AgentBody, perceptionListener: Agent)
4   |   def influence(body: AgentBody, influences: Influence*)
5   |   def destroyBody(body: AgentBody)
6   | }
7   | class PhysicSpaceImpl extends EventSpaceImpl
8   |     implements PhysicSpace {
9   |   val env: Environment
10  |   def influence(body: AgentBody,influences: Influence*) {
11  |     for(i: influences) emit(i, new Scope(env))
12  |   }
13  |   ...
14  | }
15  | skill RoadEnvironmentSkill implements RoadEnvironmentCapacity {
16  |   var body : AgentBody
17  |   def install {
18  |     body = bodyFactory.newInstance
19  |     getSpace(PhysicSpace).putInEnvironment( body, owner)
20  |   }
21  |   def influence(inf: Influence) { getSpace(PhysicSpace).influence(body, inf)}
22  |   def uninstall { getSpace(PhysicSpace).destroyBody(body) }
23  |   ...
24  | }
```

Script 1.3. Specification of the physical space

SARL enables to define a special type of space, called `Internet`, which gives a support to social interaction between agents.

The default interaction model proposed by the SARL language (`EventSpace` interface and its implementation `EventSpaceImpl`) uses events as support for the interaction: *a message becomes an event for agents that are receivers.* Then the communication

environment can be defined as the specialization of an event space, and that ensures the link to the environment entity.

```
space InternetSpace {
  def emit(e: Message, scope: Scope)
  def register(agent: Agent): Address
  def unregister(agentAddress: Address)
}
class InternetSpaceImpl extends EventSpaceImpl
    implements InternetSpace {
  val env: Environment
  def emit(e: Message, scope: Scope) {
    e.destination = scope
    super.emit(e, new Scope(env))
  }
}
```

Script 1.4. Internet space specification

Script 1.4 gives the definition of a communication space (`InternetSpace`) and a possible implementation (`InternetSpaceImpl`) based on the SARL tools. In the communication space, the agents communicate by message thanks to the capacity, and the skill described in Script 1.5.

```
event Message {
  var destination: Scope
}
capacity InternetCapacity {
  def emit(e: Message, scope: Scope=null)
}
skill InternetSkill implements InternetCapacity {
  def install { getSpace(InternetSpace).register(owner) }
  def emit(e: Message, scope: Scope=null) { getSpace(InternetSpace).emit(e, scope) }
  def uninstall { getSpace(InternetSpace).unregister(owner) }
}
```

Script 1.5. Internet interaction capacity

5 Model extension for traffic simulation

The two dimensions of the environment presented above are supported and combined within a unique entity (`Environment`).

```
behavior Environment {
  var roads : RoadNetwork
  var physicSpace : Space
  var communicationSpace : Space

  var rules : List<Pair<
      (Behavior, Event, Object) => boolean,
      (Behavior, Event, Object) => boolean>>

  def applyRules(e: Event, o: Object) : boolean {
    var propagate = true
    for(pair: rules) {
      if (pair.left.invoke(this, e, o))  {
            var p = pair.right.invoke(this, e, o)
          propagate = propagate && p }
    }
    return propagate
  }
  on Influence {
    if (applyRules(occurrence, occurrence.object)) {
      saveInfluenceForEvolve(occurrence)      }
  }
  on Message {
    for(participant: communicationSpace.participants) {
      if (occurrence.scope.matches(participant)) {
```

```
26          if (applyRules(occurrence, participant)) {
27            saveMessageFoEvolve(occurrence)
28          }   } }
29    }
30    on Initialize {
31      /* Rule for case a */
32      rules += [env,e,o | e instanceof  PriorityRequestMessage] =>
33        [ env,e,o | e.scope = Scopes.addresses( env.roads.vehiclesAtDistance( e.source, env.
              physicSpace.V2X_distance)) ]
34      /* Rule for case b */
35      rules += [ env,e,o | e instanceof Siren] =>
36        [ env,e,o | env.communicationSpace.emit( new PriorityRequestMessage(e.source)) ]
37      rules += [ env,e,o | e instanceof PriorityRequestMessage] =>
38        [ env,e,o | env.physicSpace.influence(  new Siren(e.source)) ]
39      /* Rule for case c */
40      rules += [ env,e,o | e instanceof PriorityRequestMessage] =>
41        [ env,e,o | env.physicSpace.influence(new PriorityRequestInfluence(e.source),Scopes.
              addresses(env.roads.rsuNear(e.source))
42          true ]
43      /* Create spaces */
44      physicSpace = currentContext.createSpace(PhysicSpaceSpecification, UUID.random, this)
45      communicationSpace = currentContext.createSpace(InternetSpaceSpecification, UUID.random,
              this)
46      /* Create the model of the physical dimension */
47      roads = new RoadNetwork(physicSpace)
48    }
49  }
```

Script 1.6. Environment behavior

In Script 1.6 (line 6), the set of combination rules is defined by the attribute `rules` as a list of pairs, where the left members are the syntactical closures with the following parameters: (i) the abstraction related to each algorithm with a dynamic behavior, including agents; (ii) the event appearing in one of the dimensions of the environment; and (iii) the object related to the event. The syntactical closures are functions returning a Boolean value that is true if the predicate (the rule) can be triggered. The right members are the closures with the same formal parameters as the left member, processing the task related to the rule and returning the Boolean value true if the propagation of the influence or the message in original dimension is authorized.

The `applyRules` function, the `on Influence` and the `on Message` event handlers in Script 1.6 (lines 10–29) represent the application of rules resulting from the combination of different environments. This arbitration heuristic consists in triggered the first matching rule, and move this rule at the end of the list for giving a chance to be trigger to another rule. The handling functions for the `Influence` (line 19) and `Message` (line 23) catch the influences in the physical dimension and the messages in the communication dimensions for applying rules on them.

The script 1.6 (lines 32–42) gives the definition of the rules related to the cases a, b and c of our illustrative example (see Section 2). The SARL notation `[ parameters | statements ]` allows defining the syntactical closures associated with the rules. And the notation `p => f` defines a rule as a pair of a predicate p and a definition f of the state changes in the environment. Rule a (line 32) permits restricting the set of receivers — by setting the scope of the event — of every priority request message to the vehicles that are close to the source of the priority request in the physical dimension of the environment. In other words, when the message `PriorityRequestMessage` is received, its scope is set to the vehicles that are close — according to the V2X propagation distance — to the source vehicle in the physical dimension. Rules b (lines 35 and 37) correspond to an interaction that is shared between the two dimensions. The first (resp. second) rule for the case b permits to emit automatically the `PriorityRequestMessage` message

(resp. `Siren` influence) in the communication (resp. physical) dimension when the agent has sent the `Siren` influence (resp. `PriorityRequestMessage` message) in the physical (resp. communication) dimension. Consequently, when an event was sent in a specific dimension, it is automatically sent in the other dimension without change of its content. The rule c (line 40) describes one example of an interaction in a dimension generates interaction in the other dimension.

In our example (illustrated by Figure 1), the regulation of the interactions between the dimensions of the environment following these rules influences the behavior of the agents. The agent A broadcast a priority request that is received by the agents O_{12}, O_{13} thanks to the rule a (line 32) even if they do not belong to the community. The agent C_{12} receives twice the message thanks to the rules a, and the Internet since he belongs to the community. Using the content of the message, the agent O_{12} adapts its behavior because it enters into the junction contrary to the agents O_{13} and C_{12}, who leave the junction. Thanks to the interaction model related to each dimension of the environment, the agent A interacts with other agents in each dimension. In the physical dimension, its siren is, for instance, perceived by the agents O_{22}, O_{14}. Without any other information, they slow down. It is the correct behavior for the agent O_{22} but not for the agent O_{14}. In the communication dimension, the agent C_{21} receives the message by the Internet and adapts its behavior following its position according to the junction and the advice given by the message.

In that way, it is possible to simulate complex situations like unexpected slowdowns and the related risks. For instance, the vehicle O_{14} slows while the vehicles C_{12} and O_{13} will not. Thanks to the rule c (line 40), the interaction in the communication dimension modifies the physical dimension with the modification of the traffic signal plan. The vehicle O_{11} will be stopped even if it receives no information coming from any dimensions of the environment.

6 Related Works

Our inspirations for the physical environment are the models for the simulation of crowds and traffic into virtual environments [5,18]. The Artifact [12], CArtAgO [14] and smart object [19] models are also an inspiration. They propose similar interaction models between agents and objects in the environment, and the definition of the latter.

The problems related to the interaction between an agent, and the physical environment have been treated with different perspectives. One of the models used in our approach is the Influence-Reaction model [8]. It supports the simultaneity of the actions in an environment by considering the interactions initiated by agents as uncertain, and detecting and resolving the conflicts between the interactions. This approach can be compared to the concept of artifact [13], which proposes to model the objects in the environment. They provide a set of actions that can be applied on each of them. A similar model named smart object is proposed for virtual environments [19]. This vision is supported by our model due to events such as `Siren`. The influences related to spatial travel (`MotionInfluence`) and those dedicated to trigger actions (`Siren`) are distinct for enabling a detailed specification of the parameters for each of them. The IODA model and its extension PADAWAN [10] allow modeling the interactions between the

agents and the various dimensions of the environment by assuming that every entity is an agent. Our model is partially incompatible with this vision in the context of the physical dimension modeling. Indeed, the bodies of the agents are not agents.

In the communication environment models, the environment is a shared space in which agents drop off or withdraw filters that describe the context of their interactions, in order to manage their multiparty communications [1,17,21]. These filters are managed by the environment. The physical environment is not separated from the communication environment, and filters always involve an agent. Therefore, it is possible to treat the use case a, but not the other two cases explicitly.

Several organizational approaches consider the environment [4,6,11]. In the context of this paper, the key element is the introduction of the concept of space as an abstraction for organizational groups and spatial areas. However, these models do not explicitly propose to consider the physical and communication dimensions jointly, as well as their direct interactions.

7 Conclusion and Perspectives

In this paper, a model for the combination of the physical and communication dimensions of the environment is proposed. It enhances the modeling capabilities of the environment, and provides the tools to define more complex behaviors to agents. The two dimensions of the environment are defined with interaction spaces in the SARL agent-oriented language. These spaces can be considered as points of view on the environment that is combining the different dimensions. The use of rules provides a general and adaptable tool for different classes of applications. Several concrete definitions of rules for the simulation of traffic are proposed. We think that the social dimension of the environment could be also supported by our model.

A perspective of our work is to relate, map and compare our model to other approaches for modeling the environment: artifacts [13], smart objects [19], holarchies [16], etc. The SARL language should be adapted for facilitating the definition and the description of the environment instances and their rules, like GAML has already done [7].

References

1. F. Badeig, F. Balbo, and S. Pinson. A contextual environment approach for multi-agent-based simulation. In J. Filipe, A. L. N. Fred, and B. Sharp, editors, *ICAART 2010 - Proceedings of the International Conference on Agents and Artificial Intelligence, Volume 2 - Agents, Valencia, Spain, January 22-24, 2010*, pages 212–217. INSTICC Press, 2010.
2. N. Bhouri, F. Balbo, and S. Pinson. An agent-based computational approach for urban traffic regulation. *Progress in AI*, 1(2):139–147, 2012.
3. A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Handheld and ubiquitous computing*, pages 304–307. Springer, 1999.
4. J. Ferber, F. Michel, and J. Baez. Agre: Integrating environments with organizations. In *Environments for Multi-Agent Systems*, volume 3374 of *Lecture Notes in Computer Science*, pages 48–56. Springer Berlin Heidelberg, 2005.

5. S. Galland and N. Gaud. Holonic model of a virtual 3D indoor environment for crowd simulation. In *International Workshop on Environments for Multiagent Systems (E4MAS14)*. Springer, May 2014.
6. A. Gouaïch and F. Michel. Towards a unified view of the environment(s) within multi-agent systems. *Informatica*, 29(4):423–432, may 2005.
7. A. Grignard, P. Taillandier, B. Gaudou, D. A. Vo, N. Q. Huynh, and A. Drogoul. GAMA 1.6: Advancing the art of complex agent-based modeling and simulation. In *16th International Conference on Principles and Practices in Multi-Agent Systems (PRIMA)*, volume 8291, pages 242–258, Dunedin, New Zealand, 2013.
8. F. Michel. The IRM4S model: the influence/reaction principle for multiagent based simulation. In *Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS07)*. ACM, May 2007.
9. J. Odell, H. Parunak, M. Fleisher, and S. Brueckner. Modeling Agents and their Environment. In *Agent-Oriented Software Engineering III*, volume 2585 of *Lecture Notes In Computer Science*, N.Y. (USA), 2002. Springer-Verlag.
10. S. Picault, P. Mathieu, and Y. Kubera. PADAWAN, un modèle multi-échelles pour la simulation orientée interactions. In *JFSMA*, pages 193–202. Cépaduès, 2010.
11. M. Piunti, A. Ricci, O. Boissier, and J. Hübner. Embodying organisations in multi-agent work environments. In *IEEE/WIC/ACM Int. Conf. on Web Intelligence and Intelligent Agent Technology (WI-IAT 2009)*, Milan, Italy., 2009.
12. A. Ricci, A. Omicini, and E. Denti. Activity theory as a framework for mas coordination. *LNCS on Engineering Societies in the Agents World III*, 2577:96–110, 2003.
13. A. Ricci, M. Viroli, and A. Omicini. Programming MAS with artifacts. In *International Workshop on Programming Multi-Agent Systems*. Springer Verlag, July 2005.
14. A. Ricci, M. Viroli, and A. Omicini. CArtAgO: A framework for prototyping artifact-based environments in MAS. In *E4MAS*. Springer Verlag, May 2007.
15. S. Rodriguez, N. Gaud, and S. Galland. Sarl: A general-purpose agent-oriented programming language. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 103–110, Aug 2014.
16. S. Rodriguez, V. Hilaire, N. Gaud, S. Galland, and A. Koukam. *Holonic Multi-Agent Systems*, chapter 11, pages 238–263. Self-Organising Software From Natural to Artificial Adaptation - Natural Computing. Springer, first edition, Mar. 2011.
17. J. Saunier, F. Balbo, and S. Pinson. A formal model of communication and context awareness in multiagent systems. *Journal of Logic, Language and Information*, pages 1–29, 2014.
18. G. Tamminga, P. Knoppers, and H. Lint, van. Open traffic: a toolbox for traffic research. In *3nd International Workshop on Agent-based Mobility, Traffic and Transportation Models, Methodologies and Applications (ABMTRANS14)*. Springer, June 2014.
19. D. Thalmann and S. R. Musse. *Crowd simulation*. Springer, 2007.
20. D. Weyns, A. Omicini, and J. Odell. Environment as a first-class abstraction in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):5–30, Feb. 2007. Special Issue on Environments for Multi-agent Systems.
21. M. Zargayouna, F. Balbo, and S. Haddad. Data driven language for agents secure interaction. In M. Dastani, A. E. Fallah-Seghrouchni, J. Leite, and P. Torroni, editors, *Languages, Methodologies, and Development Tools for Multi-Agent Systems, Second International Workshop, LADS 2009, Torino, Italy, September 7-9, 2009, Revised Selected Papers*, volume 6039 of *Lecture Notes in Computer Science*, pages 72–91. Springer, 2009.