A Unified Model for Physical and Social Environments

José-Antonio Baez-Barranco, Tiberiu Stratulat, Jacques Ferber

A Unified Model for Physical and Social Environments

José-Antonio Báez-Barranco, Tiberiu Stratulat, and Jacques Ferber

LIRMM, Univ. Montpellier 2, CNRS
161 rue Ada, 34392 Montpellier Cedex 5, France
{baez,stratulat,ferber}@lirmm.fr

Abstract. The AGRE model proposed by Ferber *et al.* is based on an interesting generalization of both physical and social environments. In this paper we revisit the AGRE model and extend it with richer social concepts such as powers, norms and a dependency relationship which is similar to the *count as* operator introduced by Searle to describe the construction of social reality. Our main contribution consists in the fact that we attribute to the environment the main role in describing and controlling (social) interaction.

1 Introduction

In the area of multi-agent systems (MAS), Castelfranchi claimed [1] that *social order*, which is a (social) metaphor for the problem of coordinating the agents or organizing the interactions among them while preserving their autonomy, could be obtained by using social concepts such as norms and social control. Norms are rules describing the expected ideal behavior of an agent or of a group of agents. Social control means that the agents themselves observe the behavior of the other agents, control if it is norm compliant and act consequently. Recently, many research works [2–5] proposed models that integrate social and organizational concepts in MAS and suggested tools to implement the social metaphor. However, most of them propose *ad-hoc* solutions of how social concepts are constructed and then manipulated. For instance, there is not always very clear if the social knowledge (e.g. the obligation to do an action, the power of doing an act, the membership to an institution, etc.) is shared among agents or is represented somehow externally and independently of them.

In previous works we also have studied how to integrate in MAS the organizational concepts of group and role [6] and proposed the AGRE model [7]. The AGRE model is based on the idea that the environment could be used to represent not only the physical part of the interaction but also its social aspect. The agents interact only with the environment which will react according to agent's influences [8] and to the rules of change defined at both physical and social levels of interaction.

In this paper we present the AGREEN model which is a revisited extension of the AGRE model. Our main goal is to provide a much simpler and unified way of representing (physical and social) environments. The originality of our proposal consists in the fact that it attributes to the environment the main role in describing and controlling (social) interaction. This is the major difference when compared with other related works that use social concepts [3, 4, 9].

The AGREEN model is based on a clear separation between what an agent tries to do and the effects obtained as consequences of its acts on the environment. The architect of an agent concentrates only on describing the internal structure of the agent, that is, on the design of the decision making mechanism that allows the agent to decide what to do next. The architect of the system describes the environment as a set of rules governing the interaction, ignoring how the agents are built. The main benefit of this separation is that it guarantees the autonomy of the agents and the non-intrusive control of their behavior. The non-intrusive control is based also on a clear distinction between what an agent can do, as capabilities (or powers), and what an agent is supposed to do as deontic constraints (i.e. obligations, permissions, interdictions).

Another advantage obtained from the separation agent/environment is that the semantics of an action could be given according to two perspectives: internal (agent's point of view) and external (environment's point of view). The external semantics could be further refined according to its social or physical aspect. This is a step forward towards giving to agent communication languages a public perspective and a social semantics, as requested by the agent community [10].

In the rest of the article we revisit the AGRE model and propose the AGREEN model that enhances AGRE with social concepts such as capabilities (powers), norms and a dependency relationship which is similar to the *count as* operator introduced by Searle to describe the construction of social reality [11].

2 Social Reality and AGRE

In this section we describe some social concepts such as norms and social reality that were announced in the original paper of AGRE but which deserve more attention.

2.1 Social Reality

The work of Searle on the construction of social reality [11] is becoming very influencing on the research in agent based systems [2, 3]. The main idea is that a *social institution*, even that it has no physical support, has its own (social) reality and is constructed by mutual convention among its members on how to interpret what happens in the physical reality. Searle makes the distinction between brute facts and institutional facts. A *brute fact* represents something (true) bellonging to the physical reality (i.e. a piece of paper with a ten euros sign marked on it). An *institutional fact* is a fact that is considered to be true by collective acceptance by the members of a group or comunity of agents (i.e. money such as ten euros). According to Searle an institution is defined in terms of two types of rules: constitutive and regulative. *Constitutive rules* show how to construct the social reality by giving an interpretation to brute facts or other social facts. They have the form "X counts as Y in context C". For instance, in the money institution a piece of paper with ten euros special printings on it counts as a ten-euro banknote. Jones and Sergot [2] give a formalization of *count as*, and present the concept of institutionalized power as being the (social) capability to act in an institution. *Regulative rules* describe ideal normative situations or behaviors from the point of view of an institution.

2.2 The AGRE Model

In [7], Ferber *et al.* propose an extension of the AGR model [6] and consider an organization as being a special kind of environment. Social actions are associated with an organization, i.e. playing a role, entering and leaving a group, communicating inside a group, etc. The main ideas presented in that work concern (i) the use of both social and physical environments to describe the interaction among agents; (ii) the concept of *space* which is a generalization of the concepts of physical area and social group, introduced to partition the environment; and (iii) the concept of *mode*, which is a generalization of the concepts of physical body and social role, used to describe the agent's capabilities to influence [8] physical and respectively social environments.

However, the AGRE model presents some inconveniences. First, the generalized concepts of space, mode and institution show very well the relationship that should exist between an agent and an environment, but they remain abstract and unused. Moreover, like in AGR, there is no explicit description of the expected behavior of the agents, i.e. a role is simply a label with no other semantics. Normally we should be able to associate to a role powers and deontic constraints such as obligations, permissions or interdictions. Finally, AGRE in its original form did not take into account the ideas on social reality by Searle. What is missing in AGRE is something similar to the *count as* relationship that links together physical and social environments or more generally any two environments.

3 The AGREEN Model

In this section we show how to improve and generalize the concepts introduced initially in AGRE. That is, we propose: (i) to use only the generalized concepts of space, mode and capability, (ii) to better explain the role of the environment from the point of view of behavior control, (iii) to give more details on the role of modes as capabilities to act in an environment, and (iv) to try to generalize the relationship existing between physical and social environments.

In Figure 1 we give a simplified description of the main relationships existing between a space, a mode and an agent. In order to interact to other agents an agent will influence a space through its mode which provides controlled capabilities to act in an environment. According to what an agent is able to do in a space the environment will react to its influences. The reaction of the environment is finally the result of a more complex interaction between various objects which have the role to encapsulate the environment's state and behavior.

3.1 Spaces

As in AGRE, we borrow concepts from the object-oriented programming paradigm to specify an environment. The role of a space is to keep the information about the state of a physical or social environment. A space is for agents a sort of interaction place. Its state can change as a consequence of their influences. A space is characterized by a name and a space type as as explained below. The state of a space is given by the state

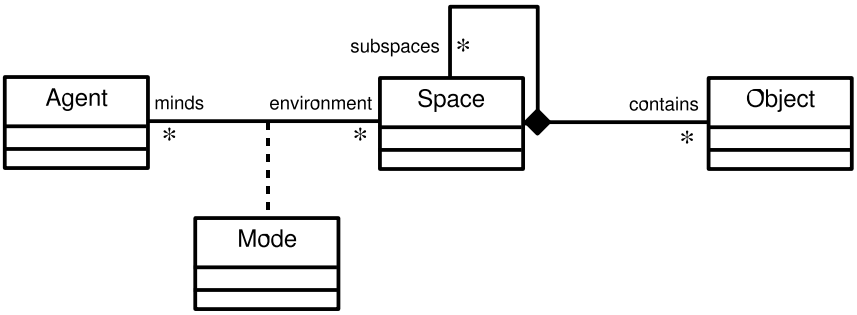

Fig. 1. Simplified UML representation of AGREEN

of the objects that compose it. The kinds of objects composing a space could be further divided in ordinary objects, modes and recursively other subspaces. In the following we will give definitions for all these concepts.

Definition 1 (Space). *A space S is a tuple <Id_S, ST, O^*, M^*, S^*> where: Id_S is the space's Id that uniquely identifies it, ST is the current space type, and O^*, M^*, S^* represent the sets of objects, modes and respectively subspaces that compose a space.*

The type of a space is a concept similar to that of class in the object-oriented paradigm and contains the description of common properties and behaviors of identical instances, i.e. concrete spaces. A space type defines the structure of its instances, i.e. the way a space instance is composed of objects, modes, and other subspaces.

Definition 2 (SpaceType). *A space type ST is a tuple <ATD_S^*, OT^*, MT^*, ST^*, DR^*, C_S^*> where ATD_S^* represents the set of attributes composing a space. An attribute could be an instance of a type taken from the sets OT^*, MT^* and ST^* which correspond to object types, mode types and respectively other space types. DR^* represents the set of explicit dependency rules that link an instance of ST to other spaces, and C_S^* is the set of environmental constraints for the environmental control.*

The set DR^* contains rules that determine how a space modifies its internal state according to some external changes produced by other spaces. A dependency rule between two spaces introduces constraints of various natures: causal (an internal event is the result of some external causes or events occurred in other spaces), logical (a local property is the logical consequence of some external properties), social (like *count as*). Note, for instance, that physical spaces have no social dependency rules.

3.2 Objects

Objects encapsulate the internal state of the environment and the laws that govern its change. An object exists at runtime and is characterized by a type and a state.

Definition 3 (Object). *An object O is a tuple <Id_O, OT, AT_O^*> where: Id_O is the object's name that uniquely identifies it; OT is the object type; and AT_O^* is the set of attributes of the object.*

The type of object is similar to the class in object-oriented programming and contains the description of common properties and behaviors of identical instances, that is concrete objects. An object type describes the possible states of its instances and how they change under agents' influences.

Definition 4 (ObjectType). *An object type OT is a tuple <ATD^*_O, M^*> where ATD^*_O is the set of attribute declarations and M^* is the set of methods defining how an object, instance of OT, changes its internal state.*

3.3 Modes

There are mainly two reasons to introduce the concept of mode: (i) to allow a space to individually attribute capabilities to agents; (ii) to allow a space to specify the expected behavior by using social deontic constraints. We propose to use the term *capability* to describe the unified concept of physical capability and social power. A capability is associated to a mode and a space, and defines the way an agent (called owner of the mode) is able to modify the space at runtime. A mode is characterized by a mode type and a set of attributes.

Definition 5 (Mode). *A mode M is a tuple <Id_M, MT, A, AT^*_M, OPI^*> where: Id_M is the mode's name that uniquely identifies it, MT is the mode type, A is the owner's (agent) identifier, AT^*_M is a set of attributes, and OPI^* is a set of deontic constraints such as obligations, permissions and interdictions.*

Definition 6 (ModeType). *A mode type MT is a tuple <ATD^*_M, P^*, C^*_M, N^*> where: ATD^*_M is the set of attribute declarations, P^* is the set of capabilities (or powers) that an instance of MT will offer to its owner, C^*_M is a set of conditions that should be fulfilled by an agent to obtain a mode in a space or to release it, and N^* is a set of norms that describe the conditions of apparition of deontic constraints that apply to a mode, instance of MT.*

The type of mode is similar to the class in object-oriented programming. It is an abstract description of the internal structure of its instances, the modes, and of the operations that could be executed on them to change their state. Since the notion of type is similar to that of class, we also introduce the concept of inheritance between two types. As in object-oriented software engineering, the inheritance is used to represent the *is-a* relationship or to reuse code.

In the definition of a mode type, the set C^*_M contains the conditions that should be verified on an agent at the creation of its mode or rechecked later to see if the agent still posses the necessary conditions to continue to interact with the environment.

The role of capability rules is to define what is possible for an agent to do in a space. When an agent influences the environment, the capability rule triggered by its mode is immediately executed by the environment. A capability rule also contains the preconditions on producing an influence on the environment. As shown before, a mode encapsulates the conditions that gives its owner the possibility to act in an environment. The capability rules are mainly employed to externally control the behavior of the agents inside a space through their modes, hence preserving their internal autonomy.

The role of norms is to implement the social control since they reflect the deontic aspect of interaction. They are of the following conditional form:

$$\textbf{if } Condition \textbf{ then } OPI(agent, \alpha)$$

where $OPI(agent, \alpha)$ describes deontic constraints such as the obligation, the permission or respectively the interdiction to do something or to arrive in a certain state of affairs α. If the $Condition$ part in a norm is $true$, the norm describes a permanent deontic constraint. We note that we do not consider general deontic constraints as in standard deontic logic, but directed deontic constraints on specific agents. Like capabilities, a deontic constraint is always connected to a mode.

As shown in the previous section, we only consider the context of social interactions. For instance, a physical mode (e.g. a body), is only a mode with an empty set of norms.

A norm could be used in various ways, for instance, as a simple container of normative information, eventually sanctioned by social penalties if violated and rewarded otherwise, or "regimented" by social mechanisms that force or block the execution of agent's actions in a space thanks to its modes. Since there is no common agreement on how to use norms in multi-agent systems, we leave open to an architect the choice of semantics and their implementation.

4 Example: modeling Warbot

In this section we illustrate the concepts introduced in AGREEN by modeling Warbot [12], a video game where two teams of robots fight against each other to destroy the opponent's base.

In Warbot there are three physical types of robots:

- $Bases$, fixed robots which are able to perceive large areas and transform food in new agents,
- $RocketLaunchers$, mobile robots that have to bring food to the base and which are able to shoot other robots perceived in their neighborhood,
- $Explorers$, mobile robots specialized in searching for food but which move faster than $RocketLaunchers$.

Warbot is constructed following the idea of separating the minds of the agents from their bodies which are situated in an environment. That is, we consider that behind each robot there is an autonomous agent which decides for the robot what to do next. The environment (or the spaces) offers the agents the capabilities to interact. Thus, the interaction could be divided in four spaces: $arena$ which is used for the physical part of the game, $lions$ and $tigers$ which correspond to the organizational aspect of the game, and $game777$ which contains the score and the entire lifecycle of one play of the game. More exactly, the agents use robots to confront physically in $arena$ space, coordinate their attacks in $lions$ and $tigers$ spaces, and play one round Warbot games in $gameXXX$ spaces. A game play is finished if one of the bases has been destroyed in the $arena$ space.

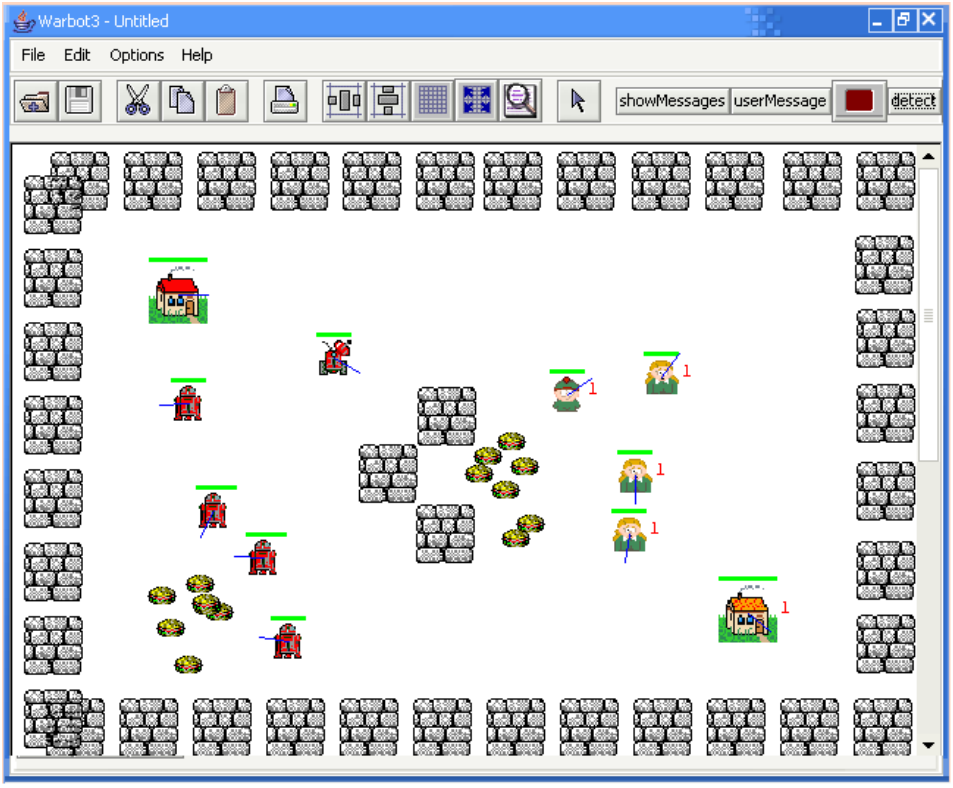

Fig. 2. Warbot, a robot video game

The space $arena$ is defined as a two-dimensional physical space allowing physical robots to move, shoot, communicate in a certain range and pick up food. The space $arena$ is an instance of $WarbotArena$, a space type which contains the definitions (see Figure 3 for some excerpts) of object types such as $Food$, $Obstacle$ and $Missile$, and mode types such as $Robot$, $MobileRobot$, $RocketLauncher$, $Explorer$, $Base$. The $Robot$ type is the super type of all the robot types and defines the capability of a robot to perceive and communicate in a certain range; $MobileRobot$ inherits from $Robot$ and adds the capability to move. Both $RocketLauncher$ and $Explorer$ inherit from $MobileRobot$. $Base$ inherits directly from $Robot$ and adds the capability to create new robots from food.

The spaces $lions$ and $tigers$ are spaces for social interaction between the members of the same team. Each team can have its own organizational structure and coordination rules, but for simplicity reasons we suppose that both teams are instances of the same space type $Team$. For instance, we can have that $Team$ is composed by one $Leader$ and many $Subordinate$s. All the modes in a space instance of $Team$ have the capability to communicate. The $Leader$ of a team contains in addition the capability (power) to give orders to a $Subordinate$. When a $Subordinate$ receives an order it becomes

```
SpaceType WarbotArena {
   ModeTypes = {
      ModeType Robot = {
         Attributes = {
            int radius = 12;
            int energy = 1000;
            int detectionRange = 130;
         }
         Capabilities = {
            int getEnergy() {...}
         }
         Conditions = {
            boolean onDemand() {...};
            boolean onRelease() {...};
            ...
         }
      }
      ModeType MobileRobot extends Robot
           {
         Attributes = {int speed;}
         Capabilities = {
            void move () {...};
         }
         ...
      }
   }
   ObjectTypes = {
      ObjectType Food {...}
      ObjectType Obstacle {
         int x, y;
         int getX(); int getY();
      }
}
```

```
SpaceType Team {
   ModeTypes = {
      ModeType Leader {
         Capabilities = {
            void order(Subordinate ag,
                 Request r) {...}
            boolean onDemand() {
               return plays(this.owner,
                    MobileRobot,
                    WarbotArena) &&
                    leader == null;
            }
         }
      }
      ModeType Subordinate { ...
         Norms = { {
            if ordered(Leader l, this,
                 act)
            then obliged(this, act, l)},
                 ...
}

SpaceType WarbotPlay {
   ModeTypes = {
      ModeType Player { ...
         Norms = { {
            forbidden (this, nextTo(this,
                 base, 10)) }
   ...
   Dependencies  = {
      { WarbotArena,
         // count as
         boolean nextTo(Player p, Base b,
              int time) {...}
      }
   }
}
```

Fig. 3. Types definitions for spaces, modes and objects

obliged to obey the order. This rule is described by a norm defined in $Subordinate$ mode type. Depending on the strategy of a team, the violation of such a norm could trigger various institutional actions, for instance, that affect the agent's trust level or socially isolate the agent from the rest of the team.

The space $game777$ is an instance of $WarbotPlay$, a space type that defines the rules of the game. It contains, for instance, the conditions to start, pause or end a game, the conditions of a team to score one point or win a game, the penalties attributed to a team if some of its members don't obey the normative rules, such as a $Player$ in a team should not stay next to a base more than a certain period of time. The constitutive rules of $WarbotPlay$ depend mainly on what happens in an instance of $WarbotArena$.

In Figure 4 we show some possible content of a space at runtime.

```
arena = {
   id = arena,
   spaceType = WarbotArena,
   subspaces = {},
   modes = {
      { id = base1,
        type = BaseRobot,
        owner = agbaseA,
        attributes = {radius = 20, energy
             = 2000, ...},
        deontic constraints = {},
      },
      { id = explorer1,
        type = Explorer,
        owner = rob33,
        attributes = {..., x = 10, y =
            23, speed = 20, ...},
        deontic constraints = {}
      },
      ...
   },
   objects = {
      { id = ob1,
        type = Obstacle,
        attributes = {x = 20, y = 30}},
      },
      ...
   }
}
```

```
lions = {
   id = lions,
   spaceType = Team,
   subspaces = {},
   modes = {
      { id = boss,
        type = Leader,
        owner = ag332,
        attributes = {ordered(ex1,
            returnHome)},
        deontic constraints = {},
      },
      { id = ex1,
        type = Explorer,
        owner = rob33,
        attributes = {...},
        deontic constraints = {obliged(
            returnHome, boss)}
      },
      ...
   },
   objects = {...}
}
```

Fig. 4. Spaces at execution time

5 Conclusions

The AGREEN model described in this paper is a revisited extension of the AGRE model. Its main goal is to provide a much simpler and unified way of representing (physical and social) environments. The model is based on: (i) a clear separation between what an agent tries to do and the effects obtained as independent consequences of its acts on the environments and (ii) a clear distinction between what an agent can do, as capabilities, and what an agent is supposed to do, as deontic constraints.

Another message of this article is that, the institution, defined by Searle as a set of constitutive rules of the form "X counts as Y in context C", is a concept general enough to expressively describe mediated interaction and environment-based coordination. In AGREEN, the space and its type, taken together, should be seen as the basic institutional unit. A space represents that part of the social reality constructed according to the (constitutive) rules defined by its type. A space type actually corresponds to the context C in Searle's terminology or to S in Jones and Sergot's formalization of *count as* operator (e.g. $X \Rightarrow_S Y$). It regroups all the constitutive rules relative to the same context C. When modeling the interaction, the expressive power of institutional concepts comes from the fact that we can divide the whole space of interactions in smaller parts and consider them separately, in isolation or connected to others.

The unified institutional model proposed in this article, allows someone to uniformly describe various types of interaction. The difference between physical and social interaction is that in the case of a physical space we don't have to specify deontic constraints and social dependency relationships. We note, however, that the social dependency relationship, which is similar to the *count as* operator, and the more general notion of dependency between any two spaces deserve both more attention and formal definitions. This will be subject of our future work.

Acknowledgements

This work has been supported by France Telecom R&D. We would like to thank Ludivine Crepin, Robert Demolombe, Vincent Louis, David Sadek and John Tranier for the fruitful discussions on various aspects described in this article, and the anonymous reviewers for their valuable suggestions.

References

1. Castelfranchi, C.: Engineering social order. In: ESAW'00. Volume 1972 of LNCS. (2000)
2. Jones, A.J.I., Sergot, M.: A formal characterisation of institutionalised power. Journal of the Interest Group in Pure and Applied Logics **4**(3) (1996) 429–445
3. Boella, G., van der Torre, L.: Structuring organizations by means of roles using the agent metaphor. In: WOA'04. (2004)
4. Vázquez-Salceda, J., Dignum, V., Dignum, F.: Organizing multiagent systems. Autonomous Agents and Multi-Agent Systems **11**(3) (2005) 307–360
5. Stratulat, T., Clérin-Debart, F., Enjalbert, P.: Norms and time in agent-based systems. In: Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL'01), St. Louis, Missouri, USA (2001)
6. Ferber, J., Gutknecht, O.: Alaadin: a meta-model for the analysis and design of organizations in multi-agent systems. In: ICMAS'98, IEEE Computer Society (1998) 128–135
7. Ferber, J., Michel, F., Báez-Barranco, J.A.: AGRE : Integrating environments with organizations. In: Environments for Multi-agent Systems. Volume 3374 of Lecture Notes in Computer Science., Springer (2005) 48–56
8. Ferber, J., Müller, J.P.: Influences and reaction : a model of situated multi-agent systems. In: Proceedings of the 2nd International Conference on Multi-agent Systems (ICMAS-96), The AAAI Press (1996) 72–79
9. Artikis, A., Pitt, J., Sergot, M.: Animated specifications of computational societies. In: AAMAS '02, New York, NY, USA, ACM Press (2002) 1053–1061
10. Singh, M.P.: Agent communication languages: Rethinking the principles. Computer **31**(12) (1998) 40–47
11. Searle, J.R.: The Construction of Social Reality. The Free Press (1995)
12. Warbot: A robot video game. http://www.warbot.org (2005)