Oversensing with a Softbody in the Environment
Another Dimension of Observation

Eric Platon and Shinichi Honiden
National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda, 101-8430 Tokyo, Japan
{platon, honiden}@nii.ac.jp

Nicolas Sabouret
Laboratoire d'Informatique de Paris 6, 8, Rue du Capitaine Scott, 75015 Paris, France
nicolas.sabouret@lip6.fr

Abstract

Research in Multi-Agent Systems extensively approached observation mechanisms *from outside*. Agents are *black boxes* and much work allows inferring internals or collective activities from the observation of their interactions. In this paper, we propose to extend the expressiveness of software agents and enact new observable features.

Our framework leads to an extended definition of agent; the central role of a software *environment* to deal with these agents; and a mechanism we named *oversensing* to refer to the observation of these agents. This paper presents these notions and provides preliminary insights on their exploitation.

1 Introduction

Modelling others from observation aims at representing others internal states or intentions, so that the observer can react accordingly. In the context of Multi–Agent Systems (MAS), agents are considered as 'black boxes' that encapsulate and hide internals from outside observation. Despite the encapsulation barrier, some internals can be inferred indirectly by observing agents interactive behaviours and applying recognition mechanisms [Kaminka *et al.*, 2004; Mitchell *et al.*, 1994]. However, observing interactions and applying recognition algorithms can be costly and error–prone. The 'indirect deduction' of internals they yield can be biased and lead to wrong decisions.

In this paper, we propose a framework to improve the observation of software agents in such a context. To this end, we think of a *software agent* not only as an autonomous and interactive entity but also endowed with an explicit boundary that exposes a selection of internals. Such an extension opens new possibilities for agent observation by expanding expressiveness, and we will define *oversensing* as a mechanism to perform such observations. In addition, we introduce an explicit software environment as first–class entity in MAS to regulate the framework exploitation and the execution of oversensing.

In section 2, we motivate in further details this research. Section 3 is devoted to define key concepts exploited in this paper. Section 4 presents how our framework participates in expanding the information that can be observed about agents. Section 6 discusses this paper and refers to related work, and finally section 7 concludes.

2 Motivations

Much research deals with observing interactions to deduce or validate hypothesis on agent internals or their collective behaviours. For instance, statistical tools applied at runtime or to log files can show trends in the interactivity of agents, either software, human, or robots [Legras and Tessier, 2003; Sabouret and Sansonnet, 2002; Mitchell *et al.*, 1994]. Monitoring MAS with *overhearing*[1] enacted non–intrusive system surveillance, conversation recognition, or agent dynamic coordination [Kaminka *et al.*, 2002; Gutnik and Kaminka, 2004; Legras and Tessier, 2003].

However, some scenarios involving software agents require other observation techniques, and we illustrate an example hereafter. In a simple settings of load–balancing, agents A and B process incoming requests received from clients. If for some reason A is overloaded and B becomes idle, it would be pertinent to share the load with B. However, this is usually addressed with a specific interaction protocol between A and B. As A is overloaded, the preparation of such a protocol is too costly. An alternative solution would be for B to recognise that A is overloaded, so that B can 'prehend' tasks. This might also be costly as B has to spend time and power on performing the observation algorithm. In addition, recognition might lead to erroneous deduction ('A is fast' instead of 'A is busy'), and consume even more time. As an alternative, we propose in this paper a framework where A shows her busy state explicitly, so that B can simply 'read' it to engage possible support.

In this example observation appears opportunistic in the sense that the observer agent engaged his role suddenly, out of context. This class of observable events is unpredictable, but if software agents can expose 'bootstraps' about their internals, like robots with their body or humans with their emotions, agent interactions and their applications are significantly extended.

[1] *Overhearing* is an indirect communication type that allows agents 'listening to conversations' without the status of addressee [Dignum and Vreeswijk, 2004]. *E.g.* one can listen to the discussion between two friends without being part of the exchange.

3 Definitions

In this section, we present definitions of agent and environment that fit our argument and we introduce *oversensing* that augments software observation.

3.1 Software Agent

In the frame of this paper, we focus on software agents. Although accurate definitions were already given [Ferber, 1999; Russel and Norvig, Edition 2003], we propose a version that emphasises agent interactions

> A software agent is an autonomous problem–solving core endowed with an explicit boundary that exposes:
>
> **sensors** to receive information from the environment
>
> **actuators** to send information to the environment
>
> **a public state** of agent internals observable in the environment

First, this definition separates agent internals from their interactive capabilities. These ones are gathered in an 'explicit boundary', in the way robots have a body shell with sensors and actuators. Second, the usual sensors and actuators are extended with a *public state*. The public state is intended to expose some agent internals as observable features. Others can observe the public state at a low cost since they can simply 'read' it.

A formalisation of this agent definition should then feature the two parts of an agent, together with their relations as a whole:

$$Agent = (\psi, \varphi, INF) \tag{1}$$

First in this formula, ψ denotes the problem–solving abilities of the agent (it's 'mind'). Second, φ is the 'body' of the agent. It refers to what we call the software body or *softbody*. This second part of the agent architecture is a 3–tuple:

$$\varphi = (\mathcal{S}, \mathcal{A}, \mathcal{P}_s) \tag{2}$$

where $\mathcal{S}$ is the set of sensors of the agent, $\mathcal{A}$ the set of actuators, and $\mathcal{P}_s$ the *public state*, which can be represented by a list of variables.

The last element of the agent definition is the INF operator (we will read it 'influence') for the agent to act on its softbody and modify its state:

$$\begin{array}{rccl} INF: & \Psi \times \Phi & \rightarrow & \Psi \times \Phi \\ & (\psi, \varphi) & \mapsto & (\psi', \varphi') = INF(\psi, \varphi) \end{array} \tag{3}$$

From an initial state of the softbody φ and the mind ψ, an evolution occurs when ψ influences φ toward another state pair: $(\psi', \varphi') = INF(\psi, \varphi)$.

In the agent community, definitions usually refer to ψ with sensors and actuators. The introduction of a softbody φ with the public state extends this model so that the distinction software agents, physical robots, and human–beings is reduced. For this reason, we will now refer to software agents simply as *agent*.

3.2 Environment

Our definition of agent lets them free to fake attitudes and express any kind of $\mathcal{P}_s$, so that observation is not guarantee to be reliable. This is the reason why the environment has a central role in definition 3.1 to isolate and structure agents in MAS. The environment should manage and regulate the performance of interaction in the system, so that agents have to comply with system rules and cannot display fake states.

Our definition is based on Weyns *et al.* from [Weyns *et al.*, 2005] and it focuses on the management of agent interactions.

> An environment is a software entity in which agents exist and that:
>
> - Structure the system
> - Mediate all interactions
> - Define and enforce interaction rules

An environment defines the structure of the system where agents evolve. This evolution is mostly interaction and it is mediated by the environment, so that it can enforce environmental rules and perform specific mechanisms, such as the one we will define for oversensing.

To formalise this definition, we propose the following:

$$Environment = (\Omega, \Phi, TRANS) \tag{4}$$

First, Ω is a 2–tuple state representing the environment internals:

$$\Omega = (\mathcal{T}opology, \mathcal{R}ules) \tag{5}$$

where $\mathcal{T}opology$ describes the environment infrastructure in terms of agents and their relations. In addition the topology defines a 'logical scope' for each agent that determines the 'closeness of agents' in the system. The scope of agents indicates what they can observe in the environment and which other agents can observe them. $\mathcal{R}ules$ is the policy that governs this environment. Rules define how the environment manages the execution of agent interactions that can be either direct, indirect, or they can also state 'laws of Physics' in some applications (*e.g.* into water).

Φ is the set of softbodies to represent the population of agents managed by the environment. Only softbodies are required since they are the observable part of agents. They are shared interface between agent and environment whereby the environment can uphold rules and ensure no incoherent or fake attitudes can occur. The environment is an enforcing infrastructure that can be tailored for specific observational situations.

In order to show the influence of the environment on agents, we introduce the $TRANS$ operator (we will read it 'transform').

$$\begin{array}{rccl} TRANS: & \Omega \times \Phi & \rightarrow & \Omega \times \Phi \\ & (\omega, \varphi) & \mapsto & (\omega', \varphi') = TRANS(\omega, \varphi) \end{array} \tag{6}$$

From Ω and any softbody φ, $TRANS$ produces a new softbody φ' and environment state Ω' that represent the evolution of φ under the influence of the environment. We can imagine it as a state transition of the softbody constrained by environmental rules and topology.

3.3 Oversensing

We define oversensing in the context of the previous sections by the following:

> Overhearing is an ensemble of techniques devoted to express and sense the environmental contribution of other agents through their public states.

Oversensing then refers to mechanisms whereby agents exploit public states. The next section stems from this definition to detail these mechanisms, focusing on the static expression of the public state and the environmental influence.

4 Oversensing Observation

4.1 Oversensing Types

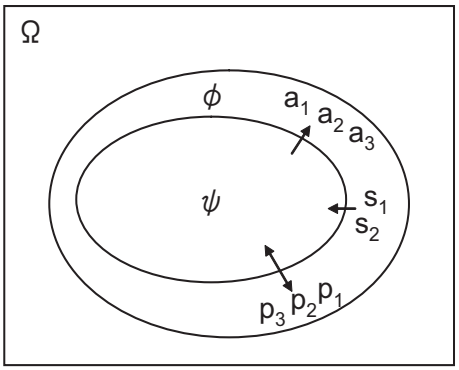

Figure 1: Agent in the environment Ω: Sensors s_i sense Ω, Actuators a_i influence Ω, and the public state $\mathcal{P}_S$ represents the agent.

Oversensing can be conscious in the case where an observer is searching for information, and we call it active oversensing. A simple algorithm can be designed to take the public state into account in observation conducted by agents.

Algorithm 1 Observation Steps of an agent (active oversensing)

1: $x \leftarrow$Results of observing interactions
2: $y \leftarrow$Results of observing public states
3: $z \leftarrow$Summarise results (x, y)
4: Return z

All potential sources to be observed are represented on Fig. 1: actuators and sensors of agents that participate in interactions, and the public state. In algorithm 1, interactions are first observed on line 1, then the public states of interest on line 2, and the acquired knowledge is merged on line 3 before the agent uses them.

Other situations can be encountered without this will to observe, as they can be indirect and opportunistic. We can call them passive oversensing. Work done on overhearing already refers to this possibility when one is listening to the radio and can catch relevant information [Busetta *et al.*, 2002] or when one receives unexpected messages about activities in the system [Balbo and Pinson, 2001], and oversensing is tailored to extend these results. Designing an algorithm in this case is not straightforward. In fact, opportunistic interactions can occur unpredictably and no agent is aware of such a situation until one realises she is observer or observed. Therefore the algorithm cannot only rely on agents and the environment shall be part of it. This will be the theme of the next section.

4.2 Passive Observation and Interaction Trigger

Figure 2 depicts a communication process for oversensing–based observation. In this context, the term 'communication' should be understood as a mere information transfer.

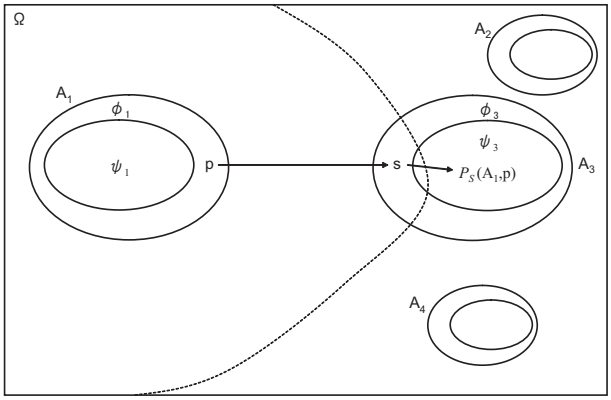

Figure 2: The environment informs agents about the public state of each other in a given range. A_3 is the only one to feel A_1 in this configuration.

A_1 exposes p as public state, for instance "Sleeping mode". The dashed line represents the range of observability of A_1 defined in the topology and whereby any agent is influenced by the environment about the state of A_1. In the situation represented on figure 2, A_2 and A_4 are out of range from A_1 and consequently are not sensitive. However, A_3 is in the scope and the environmental process is performed. With the notation introduced in section 3, we write for A_3:

$$(\Omega', (\mathcal{S}_3 \leftarrow p, \mathcal{A}_3, \mathcal{P}_{s3})) = TRANS(\Omega, (\mathcal{S}_3, \mathcal{A}_3, \mathcal{P}_{s3})) \quad (7)$$

A sensor on the softbody of A_3 is updated with p in the assignment $\mathcal{S}^3 \leftarrow p$. Once sensed this information can reach A_3's mind. On figure 2 the knowledge of A_3 is completed with the fact $\mathcal{P}_s(A_1, p)$, *i.e.* 'A_1 expresses the state p'. Such a mechanism can be illustrated by algorithm 2, distributed over the various participants.

The environment has the central role to get the expression of public states (from the update on line 2) and to spread it to other agents according to propagation rules (lines 4 to 8). However, the environment is just the medium of delivery and it has no responsibility regarding the *effect* of what it carries. In our example, p is mediated to A_3 and presented to its sensors. Depending on the configuration of A_3, the information sensed can be kept, discarded, or simply ignored.

In the case where A_3 accepts the incoming information and takes it into account, a reaction can occur. A_3 may inform A_2 and A_4 about this fact, wake up A_1 for some reason, or

Algorithm 2 Passive Oversensing Principle

Part for agent with a new $\mathcal{P}_s$ (A_1 in the example)
  Execute update procedure: $(\psi', \varphi') = INF(\psi, \varphi)$
Part for the environment
  Validate updates: $(\Omega', \varphi'') = TRANS(\Omega, \varphi')$
  if φ'' valid
    Spread in the neighbourhood
  if φ'' not valid
    Cancel the update
Part for agents in the neighbourhood (A_3 in the example)
  Read new information delivered by the environment

act while it is sleeping. These potential scenarios are examples of additional opportunistic interactions that can be spontaneously triggered in agents.

4.3 Public State Management

The softbody is the place where the public state of an agent is expressed in the environment. Its management is shared between the agent and the environment in a balanced way.

Agent and environment can influence the softbody in different ways such as conscious moves the agent wants to perform, environmental moves when a water stream carries someone, or generally the combination of both effects. However, forces are unequal. Any action from the agent is counter–balanced to comply with the 'universal' rules of the environment, so that the agent is not completely free to act. Typically, talking under water is not possible due to the environment, even if the agent wants it. Environmental effects have an overwhelming strength, and the agent can only partially balance them. Figure 3 shows the interplay on the softbody.

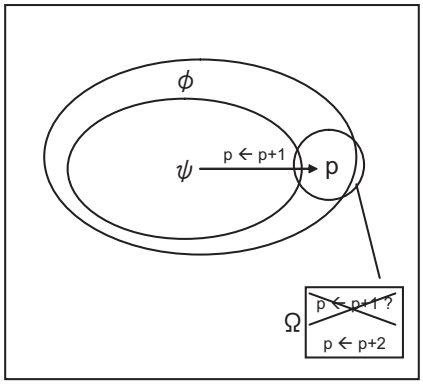

Figure 3: When agents want to update their public state (p←p+1), the environment rules the transaction and balances it according to its local laws (the effective change becomes p←p+2).

Any action denoted by an arrow from the ψ of the agent is filtered by the environment before the actual modification of the public state p. Conversely any action emitted by the environment is applied directly on the state p, although the agent can compensate the effect afterwards by another action. With our notation, the management of the softbody is conducted as follows.

$$\begin{array}{l} INF(\psi, (\mathcal{S}, \mathcal{A}, \mathcal{P}_s)) \rightarrow (\mathcal{S}, \mathcal{A}, \mathcal{P}_{s,to\ be\ validated}) \\ TRANS(\Omega, (\mathcal{S}, \mathcal{A}, \mathcal{P}_{s,to\ be\ validated})) \rightarrow (\mathcal{S}, \mathcal{A}, \mathcal{P}'_s) \end{array} \quad (8)$$

In a given state $\varphi_{initial} = (\mathcal{S}, \mathcal{A}, \mathcal{P}_s)$, the agent mind ψ influences the softbody in the part between brackets. This influence is in turn compensated by the environmental regulation with the term Ω, so that the new softbody state $\varphi_{result} = (\mathcal{S}, \mathcal{A}, \mathcal{P}'_s)$ is produced.

When the environment acts on a softbody, the performance is however directly applied:

$$TRANS(\Omega, (\mathcal{S}, \mathcal{A}, \mathcal{P}_s)) \rightarrow (\mathcal{S}, \mathcal{A}, \mathcal{P}'_s) \quad (9)$$

The agent is required to act next if it intends to counter this influence, and the effect of this new action will still be compensated by the environment (agent reactions are always regulated to enforce the correctness of their behaviour).

5 A Scenario

Agent A has boarded a flight and is sleeping. The temperature inside the plane is rather low and when agent B from the crew walks by A, B covers A with a blanket after observing the public state of A expressing 'I am getting cold'.

This scenario can be represented in our framework by the following sequence. The public state of agent A is a variable T to represent her temperature in the interval [0..10] and a variable $is\_Shaking$ to express that A is cold. The initial state is $T = 10$ and $is\_Shaking$ is false, and we consider A displays her cold feeling for $T < 5$ and turns $is\_Shaking$ to true. For sake of space and readability, we only represent the public state in the formula for the softbody (we omit sensors and actuators). We also suppose the state of the environment is constant, *i.e.* Ω is not influenced in this example by temperature changes.

Algorithm 3 Scenario stages

1: Agent A falls asleep. $T = 10$
2: The temperature of A cools down to 5 by environmental effects:
3: $TRANS(\Omega, (T, false)) \rightarrow (T-1, false)$ is repeated five times.
4: When the temperature of A reaches 4 at the next iteration, $is\_Shaking$ becomes true:
5: $TRANS(\Omega, INF(\psi_A, (4, false))) \rightarrow (4, true)$ is attempted by the agent, where the environment has no opposition.
6: When B walks by A, she puts the blanket on her.
7: The temperature now goes up:
8: $TRANS(\Omega, (4, true)) \rightarrow (5, false)$ after the first iteration.

6 Discussion and Beyond

6.1 Present Status

Oversensing is aimed at providing more matters to observe. Pragmatic benefits remain to be evaluated against the supplementary cost in terms of storage of the public state and

process complexity with the environment. These measurements should allow comparing oversensing with other techniques for common situations, and also show the intrinsic cost of oversensing for its original features.

Considering observation of the public state, a full decision process to compile a public state from system specifications is not addressed yet. However, if the aimed system has an analogy with a physical one, a heuristic might be to rely on natural senses. In the case of artificial systems without such an analogy, some insights may be inherited from the object–oriented programming community where 'public' and 'private' scopes are applied to object attributes and methods [UML, ver 2005]. But the question that remains to be clarified is the mapping between concepts at the agent and implementation level.

The environment is a critical part in this approach to compensate agent activities in a MAS and enact faithful observations. In the case of open MAS this turn to be a natural mean to regulate the system. However, its complexity and overload needs to be evaluated and compared against other approaches. The highest maintenance cost of the environment should occur when the ratio (number of agents / frequency of 'logical movements') is close to one in significantly *large* MAS[2]. In such settings, the environment must compute for each agent the oversensing range of each neighbour and evaluate whether public states must be published. Distributed environments can support similar issues [Mamei and Zambonelli, 2004; Omicini and Zambonelli, 1999] and provide potential solutions.

The operators defined for handling the softbody structure all interactions. Regarding the complexity, influence and transformation operators differ due to their nature. Influences have no cyclomatic cost since they are mere assignments of new values to variables in the public state. However, transformations require the review of environmental rules to ensure the application is valid. The transformation cost is thus directly related and proportional to the number —and complexity— of rules.

Another point with operators is their synchrony. Influences are assignments, usually considered as atomic events, so that agent internals and body are synchronised. Transformations are also synchronised, since the execution of the corresponding interaction is 'blocked' until the environment validates it. Agent interactions are however asynchronous when considered from end–to–end. The transaction is mediated by the environment, but only the beginning and the end are synchronised with agents. In–between, the environment processes the delivery but the system is not blocked. This should allow maintaining flexibility in the system, for example in the case of Internet applications.

6.2 Related Work

Coordination artefacts were recently proposed for the development of open MAS [Ricci *et al.*, 2005]. They provide agents with a standard interface to other entities in the environment, so that coordination issues and the consequent interaction concerns are managed locally. Coordination artefacts provide information about the entity they wrap to let agent exploit them 'seamlessly', and our softbody provides similar interface. However, the softbody belongs to the agent and is tailored for agent interactions and general observation. Coordination artefacts address related issues but are not appropriate for observation since they are not coupled to agents (many agents can use the same artefact).

The View Definition Language (VDL) describes agents as a 'white box' (by opposition to the black box we presented in this paper) [Sabouret, 2002]. The complete logics and internal state of the agent is compiled in a single file that is *rewritten* along the execution. Our softbody is a configurable solution between the usual black box describing agents and VDL. We think this is a critical feature to preserve the internal integrity of agents [Gouaïch *et al.*, 2005], while leveraging the exposition of part of internal states as developed in this paper.

Finally, the Behavioural Implicit Communication (BIC) was proposed as agent interaction paradigm [Castelfranchi, 2004; Tummolini *et al.*, 2005]. With BIC the authors argue that agents can express meaning when acting, in a different way that 'non–verbal' signs, either cultural or pre–decided codes, usually referring to facial expressions. On the contrary BIC is not codified and such interactions carry meaning without language. An example given in [Castelfranchi, 2004] is about a prey that escapes her predator. She actually discloses her position and intentions just by moving. The authors explain that she does not communicate this information 'intentionally', but the predator can understand she is running away and to which direction just by observing her behaviour. Different BIC levels are defined and we think the oversensing observation covers some of them. Tummolini also exploited the term 'oversensing' in [Tummolini *et al.*, 2005] to refer to a possible extension of the concept of overhearing in the case of BIC. Our approach seems compatible and is intended to be pragmatic compared to this theory.

7 Conclusion

Oversensing is a communication and interaction mean that is not addressed in MAS to our knowledge. Even robots do not usually exploit their body that is mostly thought of as an 'inexpressive shell'. Oversensing relies on the agent logical environment and on the notion of softbody to build both a dynamic and static observation scheme. In this paper we introduced these concepts as foundation for our work.

Our ongoing endeavours on oversensing aim at refining the concepts and designing appropriate algorithms. Then, implementations of the softbody and the environment should allow validating our approach.

References

[Balbo and Pinson, 2001] Flavien Balbo and Suzanne Pinson. Toward a multi-agent modelling approach for urban public transportation systems. In *Engineering Societies in the Agent World'01*, volume 2203 of *LNAI*, pages 160–174. Springer–Verlag, 2001.

[2] Movements are logical since the geometry of the environment can be anything, especially more than 3 dimensions

[Busetta *et al.*, 2002] Paolo Busetta, A Donà, and M. Nori. Channelled multicast for group communications. In *Autonomous Agents and Multi–Agent Systems*, 2002.

[Castelfranchi, 2004] Cristiano Castelfranchi. SILENT AGENTS: From observation to tacit communication. In *Proceedings of Modeling Other Agents from Observations'04*, 2004.

[Dignum and Vreeswijk, 2004] Frank P.M. Dignum and Gerard A.W. Vreeswijk. Towards a testbed for multi-party dialogues. In *Advances in Agent Communication*, volume 2922 of *LNAI*, pages 212–230. Springer–Verlag, 2004.

[Ferber, 1999] Jacques Ferber. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, 1999.

[Gouaïch *et al.*, 2005] Abdelkader Gouaïch, Fabien Michel, and Yves Guiraud. MIC*: A deployment environment for autonomous agents. In *Environment for Multi–Agent Systems'04*, volume 3374 of *LNAI*, pages 109–126. Springer–Verlag, 2005.

[Gutnik and Kaminka, 2004] Gery Gutnik and Gal A. Kaminka. Towards a formal approach to overhearing: Algorithms for conversation identification. In *Autonomous Agents and Multi–Agent Systems*, 2004.

[Kaminka *et al.*, 2002] Gal A. Kaminka, D. V. Pynadath, and Milind Tambe. Monitoring teams by overhearing: A multi-agent plan-recognition approach. *Journal of Artificial Intelligence Research*, 17:83–135, 2002.

[Kaminka *et al.*, 2004] Gal A. Kaminka, Mathias Bauer, Piotr Gmytrasiewicz, and David V. Pynadath, editors. *Modeling Other Agents from Observations*, 2004.

[Legras and Tessier, 2003] François Legras and Catherine Tessier. LOTTO: Group formation by overhearing in large teams. In *Autonomous Agents and Multi–Agent Systems*, 2003.

[Mamei and Zambonelli, 2004] Marco Mamei and Franco Zambonelli. Self-maintained distributed tuples for field-based coordination in dynamic networks. In *Symposium on Applied Computing*, 2004.

[Mitchell *et al.*, 1994] Tom Mitchell, Rich Caruana, Dayne Freitag, John McDermott, and David Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7):81–91, 1994.

[Omicini and Zambonelli, 1999] Andrea Omicini and Franco Zambonelli. Coordination for internet application development. *Autonomous Agents and Multi-Agent Systems*, 2:251–269, 1999.

[Ricci *et al.*, 2005] Alessandro Ricci, Mirko Viroli, and Andrea Omicini. Environment-based coordination through coordination artifacts. In *Environment for Multi–Agent Systems'04*, volume 3374 of *LNAI*, pages 190–214. Springer–Verlag, 2005.

[Russel and Norvig, Edition 2003] S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Edition 2003.

[Sabouret and Sansonnet, 2002] Nicolas Sabouret and Jean-Paul Sansonnet. Learning collective behaviour from local interactions. In *International Workshop of Central and Eastern Europe on Multi-Agent Systems 2001*, volume 2296 of *LNCS*, pages 273–282. Springer–Verlag, 2002.

[Sabouret, 2002] Nicolas Sabouret. Representing, requesting and reasoning about actions for active components in human-computer interaction. Technical Report 2002-09, LIMSI-CNRS, 2002.

[Tummolini *et al.*, 2005] Luca Tummolini, Cristiano Castelfranchi, Alessandro Ricci, Mirko Viroli, and Andrea Omicini. "Exhibitionists" and "Voyeurs" Do It Better: A Shared Environment for Flexible Coordination with Tacit Messages. In *Environment for Multi–Agent Systems'04*, volume 3374 of *LNAI*, pages 215–231. Springer–Verlag, 2005.

[UML, ver 2005] Unified Modeling Language. http://www.omg.org/technology/documents/formal/uml.htm, ver. 2005.

[Weyns *et al.*, 2005] Danny Weyns, H Van Dyke Parunak, Fabien Michel, Tom Holvoet, and Jacques Ferber. Environments for multiagent systems, state-of-the-art and research challenges. In *Environment for Multi–Agent Systems'04*, volume 3374 of *LNAI*, pages 1–47. Springer–Verlag, 2005.