Multi-Agent Systems as Composition of Observable Systems

Mirko Viroli and Andrea Omicini
DEIS, Università degli Studi di Bologna
via Rasi e Spinelli 176, 47023 Cesena (FC)
{mviroli,aomicini}@deis.unibo.it

Abstract

Observation is becoming a crucial issue in the engineering of today's systems: the common practice for dealing with their complexity is to encapsulate their subcomponents abstracting away from their internal details, namely, focusing on their observable behaviour.

Starting from the framework for Observation within computer systems that we developed in [5], in this paper we study the impact of thinking about agents and multi-agent systems in terms of their observable behaviour.

Firstly, we apply our ontology for observation to agents. We claim that it can be significant to describe agents in terms of knowledge sources of such an ontology, that is, focusing on their observable behaviour.

Then, we proceed by extending the whole framework for observation so as to focus on the collaboration between observable systems, where an observation manifestation is also interpreted as an observation request for another system. This naturally lead us to model a multi-agent system in terms of agents sharing their knowledge through the continuous exchange of observation messages.

1 Introduction

The impetuous development of information technologies is rapidly changing the scenario on computer science. Theoretically, the increasing complexity of computers and computer applications is making impractical to provide for complete modellisation of their behaviour. Pragmatically, computer systems tends to be built as aggregations of components, which are often knowable only in terms of the services they ask for and offer – in other words, through their observable properties. The need for dealing with this kind of systems has emerged in a plethora of different concepts, models and mechanisms. Notions as *interface* and *encapsulation*, paradigms like the object-oriented and the component-based ones, have promoted observation to a first-class issue, by implicitly stating that complexity of computer systems could be handled only by abstracting away from the essence (in a broad sense) of the elements (like objects, or components), and focusing instead on their observable behaviour.

Agent-based technologies – which are one of the most promising approaches for building today's complex systems – are contributing to this trade-off as well. In multi-agent systems (MAS) agents are typically used as abstractions to model systems where autonomous, possibly intelligent software entities interact and access heterogeneous knowledge sources in a distributed, decentralised and unpredictable environment. Very often, the only way for characterising the behaviour of an agent, so as to make him able to work with other agents, is to know its observable behaviour abstracting away from its internal aspects. As a result, agents are very often exploited as a means to deal with the unknowability of computer-based systems. For instance, agents are viable abstractions for *legacy systems*, which might be based on a technology that is no longer mastered within an organisation, or whose inner behaviour might be no longer known – for historical reasons, typically, in principle independently of its actual complexity. In general, it is frequent that interpreting the observable behaviour of a system is the only way to make it work within new systems.

Thus, the precise characterisation of the observable behaviour of an agent, and the study of how observation could be promoted to a relevant issue in the engineering of MAS as well, seem necessary efforts for harnessing the intrinsic complexity of current and future computer systems.

In [5] we introduced an ontology and a formal framework for observation. On the one hand, the ontology aims at providing a unique assessment for concepts and mechanisms related to observation, and for comparing in a qualitative way the observation properties of seemingly diverse systems. On the other hand, the formal framework allows for a precise characterisation of the observation properties, and provides a tool for specifying the observable behaviour of a component in multi-component systems.

In this paper, we apply both ontology and formal framework to agents, modeling an agent as a knowledge source with a hidden internal behaviour, and able to manifest part of it as re-active and pro-active responses to observation requests. This kind of model for an agent leads us to focus on its observable behaviour, on how this is affected by the interactions with the environment, and on how an agent makes its internal state perceivable to the outside.

According to the definition given in [6], an agent is a software entity situated in an environment and having the properties of reactivity, proactiveness, autonomy and social ability. We show how our model for agents allows the former three properties to be taken into account.

In order to deal with agent social ability as well, we proceed by shifting the focus of the framework to the *composition* of sources, so as to model a MAS. This approach allows us to interpret an evolving MAS as a system of autonomous observable sources that deliberatively share their knowledge by continuously interacting with each other. The social ability of an agent is viewed as its disposability of manifesting a part of its knowledge and its dynamics to other agents.

In general, common concepts related to agents – such as environment, interaction and MAS evolution; and agent autonomy, unpredictability, reactivity and proactiveness – will find a new interpretation in our framework, in terms of idea related to observation.

2 Observing a System

Our ontology for observation interprets computer systems as made of three kinds of entities: *sources*, *observers* and *coordinators*. A source is a component storing some knowledge and intrinsically able to *manifest* a part of it – or of its dynamics – by delivering chunks of information in forms of asynchronous messages, called *manifestations*. Observers are those entities receiving these messages, namely, *observing* the source manifestation. Coordinators are entities able to interact with a source, so as to *condition* it to produce one or more manifestations.

Correspondingly, we model a source as the composition of two parts: (i) the *core*, representing the part of its knowledge which can affect the source's observable behaviour, and (ii) the source *configuration*, containing the specification of what manifestations the source is requested to produce, and how these can influence its future behaviour. The evolution of a source among interactions with coordinators and observers – that is, among the processing of *observation actions* – is as follows.

The source starts in an *equilibrium* state, typically with an empty configuration and a certain initial core status. At a given time, this equilibrium state can be changed – crossing some *motion states*, firing manifestations, and then returning on a new equilibrium state – by one of two events. On the one hand, a coordinator can condition the source by changing the content of its configuration. In general this is accomplished through a given language, which can possibly constrain the flexibility on how coordinators can influence the source's observable behaviour. On the other hand, the source can spontaneously change its core status, modeling either an internal modification event (an internal clock, or an event related to a local processing), or an interaction with the external environment. So, when a conditioning or an internal change occurs, the configuration of the source is *evaluated*, possibly producing some messages to be sent out, a change on the core, and a change on the configuration. Correspondingly, the source returns on a new equilibrium state.

In [5] we showed how this ontology can be satisfactorily exploited for modeling a broad set of mechanisms and concepts that can be used to support observation: client-server and publish-subscribe interactions, re-activity and pro-activity, synchronous and asynchronous communications, trigger entities, and so on. For instance, our ontology emphasises that from the observation viewpoint, (i) a *rdp* operation of Linda, (ii) a field reading in object orientation, (iii) a property getting in Java Beans and (iv) a *select* query in databases, are all related to the same kind of observation, which in our ontology is called *direct re-active observation*. Namely, the coordinator conditions the source so as to produce a message for the observer without changing its core and its future observable behaviour. Another common kind of observation is the *direct pro-active* one, where the configuration is conditioned so as to produce a message each time a given event occurs on the core – modeling e.g. event notification in JavaBeans or Java Spaces.

Then, the formal framework we introduced can be exploited to describe the observable properties of a source, and, according to this, modeling the effect of conditionings on manifestations. Formally, the observation properties of a source are described by a tuple $\langle P, C, M, J\rangle$. P is the set of *places*, modeling the current state of the core. C is the set of *configuration atoms*, or *c-atoms* for short; a multiset of c-atoms $\overline{c} \in \overline{C}$ is used to model the current state of the configuration. M is the set of messages the source can produce, while $J = \langle \mathit{eval}, \mathit{select}\rangle$ is a pair of functions called *evaluation* and *selection* function. In particular, the function:

$$\mathit{eval} \in C \mapsto (P \times P \mapsto C_\perp \times P_\perp \times \overline{M})$$

accepts a c-atom c and the current transition of place $\langle p, p'\rangle$ – called *move* – and returns a triplet $\mathit{eval}(c)\langle p, p'\rangle = \langle c^*, p^*, \overline{m}\rangle$. Here:

- c^* is the new c-atom to be inserted in the configuration updating the c-atom currently evaluated c – or in the case $c^* =\perp_C$ simply modeling the dropping of c from the configuration;
- p^* updates the current move from $\langle p, p'\rangle$ to $\langle p', p^*\rangle$ – or leaves it to $\langle p, p'\rangle$ if $p^* =\perp_P$;

- $\overline{m}$ is the (multi)set of messages to be sent.

On the other hand, the selection function:

$$\mathit{select} \in \overline{C} \mapsto (P \times P \mapsto C_{\perp})$$

accepts the current *motion*, that is the pair – current move $\langle p, p' \rangle$, current configuration $\overline{c}$ – and returns the c-atom to be evaluated $c = \mathit{select}(\overline{c})\langle p, p' \rangle$, or $\perp_C$ – which means that no c-atoms are still to be evaluated and the source can return on equilibrium.

We represent equilibrium states by elements of the set $P \times \overline{C}$ and denote these states by the symbol $p\,[\,\overline{c}\,]$ – where p is the place and $\overline{c}$ is the configuration. Motion states are elements of the set $P \times P \times \overline{C}$, denoted by the symbol $\langle p, p' \rangle[\,\overline{c}\,]$ – where $\langle p, p' \rangle$ is the current move and $\overline{c}$ is the configuration. The dynamics of the source is modeled by means of a labeled transition system with two transitions from equilibrium states to motion states, called *conditioning* (C) and *source* transition (S), one from motion states to motion states, called *evaluation* transition (E), and one from motion states back to equilibrium states, called *output* transition (O). Intuitively, from equilibrium we go to motion through either a coordinator conditioning (C) or a source spontaneous change (S), then we perform a number of evaluations (E), until we return on equilibrium (O). The semantics for these transitions is defined by the following rules.

$$p\,[\,\overline{c}\,] \xrightarrow{c}_C \langle p, p \rangle[\,c|\overline{c}\,] \tag{C}$$

$$p\,[\,\overline{c}\,] \xrightarrow{p'}_S \langle p, p' \rangle[\,\overline{c}\,] \tag{S}$$

$$\frac{\begin{array}{c}\mathit{select}(c|\overline{c})(p, p') = c \\ \mathit{eval}(c)(p, p') = \langle c', p^*, \overline{m} \rangle \\ \langle p'', p''' \rangle = \begin{cases} \langle p', p^* \rangle & \text{if} \quad p^* \neq \perp_P \\ \langle p, p' \rangle & \text{otherwise} \end{cases}\end{array}}{\langle p, p' \rangle[\,c|\overline{c}\,] \xrightarrow{\overline{m}}_E \langle p'', p''' \rangle[\,c'|\overline{c}\,]} \tag{E}$$

$$\frac{\mathit{select}(\overline{c})(p, p') = \perp_C}{\langle p, p' \rangle[\,\overline{c}\,] \longrightarrow_O p'\,[\,\overline{c}\,]} \tag{O}$$

The rule (C) simply states that conditioning causes a new c-atom c to be added to the configuration. The rule (S) moves the source to a motion state considering the spontaneous transition of the core from p to p'. The rule (E) (which is fired only after either (C) or (S)), selects one c-atom from the configuration, evaluates it, and then updates the motion state of the source and sends the manifestations. Only when no more c-atom can be selected the source returns on equilibrium due to (O), updating the move $\langle p, p' \rangle$ by the place p'.

Notice that our framework leads to a slightly constrained version of the model provided by the ontology. Conditioning can be made only by adding a new c-atom to the configuration, namely through a *registration*, and the evaluation of a configuration is actually divided into a multiplicity of atomic evaluations steps, each considering one single c-atom and eventually firing the next step. However, in [5] we showed that the framework has full expressiveness for modeling the observable behaviour of well-known models such as Linda [3], JavaSpaces [2], active databases [1] and Java Beans [4].

3 Agents as Observable Systems

The most natural interpretations for agents in our ontology is to view them either as observers, interested in receiving information about a knowledge source, or as coordinators, autonomously acting on knowledge sources so as to stimulate manifestations, possibly with some intelligent behaviour. Instead, in this paper we claim that both ontology and formal framework can be successfully exploited for modeling agents as observation sources. Typically, when studying issues related to agents' behaviour, the focus mostly concerns either intra-agent aspects – capturing the agent autonomous behaviour – or inter-agent ones – related to the social ability of the agents. Here we aim at modeling the behaviour of the part of the agent devoted at interfacing these two apparently separated worlds.

In general, modeling an agent as an observable source means to concentrate on how it makes its status – or part of it – observable to other agents, and how its observable behaviour can be affected by its autonomous activity, by other agents and the environment. Thus, this also leads us to interpret agent interactions in terms of observation-related messages, that is, incoming messages as conditionings for an observation, and outgoing messages as manifestations of an observation.

Suppose to consider a single agent as a knowledge source with specification $\langle P, C, M, J \rangle$. P – the set modeling the core of the source – represents the part of the internal state of the agent that can affect its observable behaviour as perceived by the observing environment, typically including other agents. Notice that in general, this is only a sub-part of an agent knowledge, or even some abstract view of it, which may not have an actual counterpart in the agent. This satisfies the typical need of abstraction the agent methodologies require: only a part of the agent is modeled as visible from outside, while the rest remains hidden.

C – the set of c-atoms – represents the set of requests for observation an agent can accept. In general, these are the kinds of messages an agent can receive and then decide to serve, either as soon as possible, at a given time, or never. J gives semantics to c-atoms, that is, how their evaluation af-

$$\frac{\begin{array}{c} p\,[\,\overline{c}\,] \xrightarrow{p'}_S \langle p_0, p'_0\rangle[\,\overline{c}_0\,] \\ \langle p_0, p'_0\rangle[\,\overline{c}_0\,] \xrightarrow{\overline{m}_0}_E \langle p_1, p'_1\rangle[\,\overline{c}_1\,] \quad \dots \quad \langle p_{n-1}, p'_{n-1}\rangle[\,\overline{c}_{n-1}\,] \xrightarrow{\overline{m}_{n-1}}_E \langle p_n, p'_n\rangle[\,\overline{c}_n\,] \\ \langle p_n, p'_n\rangle[\,\overline{c}_n\,] \longrightarrow_O p^*\,[\,\overline{c}^*\,] \end{array}}{p\,[\,\overline{c}\,] \xrightarrow{\overline{m}_0|\dots|\overline{m}_{n-1}}_P p^*\,[\,\overline{c}^*\,]} \qquad \text{(P)}$$

$$\frac{\begin{array}{c} p\,[\,\overline{c}\,] \xrightarrow{c}_C \langle p_0, p'_0\rangle[\,\overline{c}_0\,] \\ \langle p_0, p'_0\rangle[\,\overline{c}_0\,] \xrightarrow{\overline{m}_0}_E \langle p_1, p'_1\rangle[\,\overline{c}_1\,] \quad \dots \quad \langle p_{n-1}, p'_{n-1}\rangle[\,\overline{c}_{n-1}\,] \xrightarrow{\overline{m}_{n-1}}_E \langle p_n, p'_n\rangle[\,\overline{c}_n\,] \\ \langle p_n, p'_n\rangle[\,\overline{c}_n\,] \longrightarrow_O p^*\,[\,\overline{c}^*\,] \end{array}}{p\,[\,\overline{c}\,] \xrightarrow{c,\overline{m}_0|\dots|\overline{m}_{n-1}}_R p^*\,[\,\overline{c}^*\,]} \qquad \text{(R)}$$

Figure 1. The transition system for an agent's re-active and pro-active behaviour

fects the observable behaviour of the agent. This models the agent's logics implementing the interpretation of requests, the policy of their management, and how to manifest the internal state and its dynamics outside. It can be considered as the semantics of the agent's interface to its environment. Finally, M is the set of outgoing messages the agent can emit, defining the language of its manifestation.

The evolution of an agent through the processing of observation actions can be interpreted as follows. We start by considering the agent in an equilibrium state. In general, this is not meant to model agent inactivity, but rather the agent autonomously acting without any observable effect. During this state, the agent receives conditionings, in terms of changes on its configuration, which fire evaluation. Due to the flexibility of this mechanism, some of these requests can be re-actively served, that is, producing some message and/or some change on status of the agent, both on its knowledge and on its future behaviour. Some other requests can be actually evaluated only when some condition on the agent occurs, modeling the situation where the agent decides it is time to do so. Also, the agent can decide to reject some request.

Furthermore, it can also be the case that the agent pro-actively and autonomously interrupts its equilibrium state and manifests some observable behaviour. This is modeled as a spontaneous change on the agent's core, firing an evaluation that possibly alters the configuration, the core, and produces some manifestation.

4 MAS and Observation

In order to concentrate on the interactions between agents in a MAS, so as to highlight our vision of the agents' social ability, we move the focus outside a source. First of all, we define a transition system modeling the behaviour of an agent so as this can be perceived by its environment, by wrapping the one defined in Section 2 and hiding internal details about evaluation. We introduce two transitions, from equilibrium states to equilibrium states, thus abstracting away from the motion states an agent possibly crosses. These are respectively called pro-active (P) and re-active (R) source transition. Their semantics is shown in Figure 1.

The pro-active transition $e \xrightarrow{\overline{m}_0|\dots|\overline{m}_{n-1}}_P e'$ moves an agent from the equilibrium state e to the the equilibrium state e' producing the set of manifestations $\overline{m}_0|\dots|\overline{m}_{n-1}$. This is a direct interpretation of the agent pro-actively sending messages to other agents. The re-active transition $e \xrightarrow{c,\overline{m}_0|\dots|\overline{m}_{n-1}}_R e'$, instead, makes an agent in the state e accepting the conditioning c-atom c, correspondingly producing the manifestations $\overline{m}_0|\dots|\overline{m}_{n-1}$, and then moving to the state e'. This is an interpretation of the agent receiving the message containing the request c and then re-actively producing some outgoing messages.

On top of this transition system, we proceed by focusing on a MAS. We introduce a calculus with the following syntax:

A	$::=$	$s : p\,[\,\overline{c}\,]$	Agents
μ	$::=$	$\langle s \leftarrow c\rangle$	Messages
γ	$::=$	A	An agent of a MAS
	$\mid$	μ	A message pending in a MAS
	$\mid$	$\gamma \oplus \gamma$	a composition of MAS
	$\mid$	ϵ	an empty MAS

An agent $s : p\,[\,\overline{c}\,]$ is a source in the place $p\,[\,\overline{c}\,]$, having the unique name $s \in \mathcal{S}$. A message $\langle s \leftarrow c\rangle$ is sent to the agent named s and carries the c-atom c as content. Finally, a MAS γ is a composition of agents (supposing they have different names) and floating asynchronous messages that wait to be received.

The key aspect of our composition of agents is that each message sent by an agent – which is a manifestation of its observable behaviour – is treated, by the agent receiving it, as a conditioning for an observation. We define a computational model for MAS as a labelled transition system on γ terms, defined by the rules shown in Figure 2.

The rules [MAS-π] and [MAS-ρ] define the effects on the MAS when one of its agents performs a (P) and (R)

$$\frac{p\,[\,\overline{c}\,] \xrightarrow{\overline{\mu}}_P p^*\,[\,\overline{c}^*\,]}{\gamma \oplus s : p\,[\,\overline{c}\,] \longrightarrow_\pi \gamma \oplus s : p^*\,[\,\overline{c}^*\,] \oplus \overline{\mu}} \quad (\text{MAS-}\pi) \qquad \gamma \oplus \mu \xrightarrow{\mu}_\omega \gamma \quad (\text{MAS-}\omega)$$

$$\frac{p\,[\,\overline{c}\,] \xrightarrow{c,\overline{\mu}}_R p^*\,[\,\overline{c}^*\,]}{\gamma \oplus s : p\,[\,\overline{c}\,] \oplus \langle s \leftarrow c \rangle \longrightarrow_\rho \gamma \oplus s : p^*\,[\,\overline{c}^*\,] \oplus \overline{\mu}} \quad (\text{MAS-}\rho) \qquad \gamma \xrightarrow{\mu}_\iota \gamma \oplus \mu \quad (\text{MAS-}\iota)$$

Figure 2. The transition system for MAS

transition, respectively. In the transition:

$$\gamma \oplus A \longrightarrow_\pi \gamma \oplus A' \oplus \overline{\mu}$$

the agent A situated in the MAS γ pro-actively sends the messages $\overline{\mu}$ and turns itself into A', according to semantics of the rule [MAS-π]. In the transition:

$$\gamma \oplus A \oplus \langle s \leftarrow c \rangle \longrightarrow_\rho \gamma \oplus A' \oplus \overline{\mu}$$

the agent A with name s and situated in the MAS γ, re-actively consumes the message $\langle s \leftarrow c \rangle$, sends the messages $\overline{\mu}$ and turns itself into A', according to semantics of the rule [MAS-ρ].

Thus, each message $\langle s \leftarrow c \rangle$ produced by an agent, and containing the c-atom c, is a manifestation remaining in the MAS until the agent with name s actually consumes it. When this happens, c is considered by this agent as a conditioning. The reader should notice that independently from which pattern of message passing exists between the agents of the MAS, this can always be modeled as a sequence of (MAS) transitions [MAS-π] or [MAS-ρ].

The rules [MAS-ω] and [MAS-ι] allow messages to be respectively dropped and put in the MAS by the external environment. On the one hand, the rules [MAS-π] and [MAS-ρ] can be considered computational rules of the MAS, since they describe the semantics of the evolution of the MAS. On the other hand, the rules [MAS-ω] and [MAS-ι] describe the interactions of the MAS with its environment.

After this ”syntactic” composition of sources, now it is time to recover the corresponding meaning from the observation viewpoint. We start by focusing on the rules [MAS-π] and [MAS-ρ]. Initially, each agent has its own knowledge, only a part of which is observable by its environment. Each transition corresponds to the agent manifesting part of its knowledge to other agents, possibly (in the case of [MAS-ρ]) by itself accepting a manifestation from another agent as a conditioning. In general, the MAS evolves through a continuous exchange of manifestations between agents. As transitions occur, the knowledge of each agent tends to be shared through all the agents of the MAS.

In other words, provided that the social behaviour of agents takes place through the message they exchange, then the observation viewpoint makes it possible to interpret the social behaviour as the process of sharing individual knowledge (beliefs, desires, information), so as to grow a shared, social knowledge. The social ability of an agent, then, can be viewed as its disposability of manifesting part of its knowledge so as to contribute to the creation of the MAS shared knowledge.

Finally, the rules [MAS-ι] and [MAS-ω] makes it possible to have a certain degree of openness in the MAS. The rule [MAS-ι] models the fact that the environment of the MAS can potentially contribute in defining the global shared knowledge, by adding information via floating messages. On the other hand, the rule [MAS-ω] models manifestations not conditioning other agents in the MAS, but being consumed by the environment. In some sense, this can allows to view a MAS itself as a an observation source, receiving conditionings and sending manifestations.

5 Conclusions

In this paper we applied general concepts related to Observation within computer systems, as defined in [5], to the field of MAS.

As a first step, we modeled agents as observable sources. This is a somewhat infrequent point of view, since agents are not usually seen as knowledge sources making observable their status, but rather as autonomous components eventually cooperating with other agents. This interpretation allows us to see agent interactions as incoming requests for an observation and as outgoing manifestations of an observation. To this end, the formal framework introduced in [5] turned out to be a successful tool for specifying the observable behaviour of an agent. Its key property is to abstracts away from those issues related to an agent's internal management not affecting its interactions with the environment. Moreover, this framework also forces us to focus on one aspect that the current research seems not to systematically address, that is, the interface between the intra-agent and the inter-agent viewpoints.

Then, we extended the framework so as to focus on the cooperation of sources, by introducing a calculus for MAS based on a transition system. In the corresponding model, a MAS is seen as a computational system able to accept conditionings from the external environment, and able to manifest its internal changes. A MAS internal evolution is naturally interpreted as a continuous exchange of manifestations between its agents, leading to the share of each agent's local knowledge, so as to grow a social MAS knowledge.

We claim that observation should be promoted to a first-class issue in the modelling and the engineering of Multi-Agent Systems. As a consequence, this work is meant to pose the conceptual and formal framework for reasoning about the observation issue in the field of Multi-Agent Systems. Our future work is devoted at clearly understanding the consequences of this approach, and to apply corresponding ideas, concepts and models, to define effective engineering methodologies for Multi-Agent Systems.

References

[1] P. Fraternali and L. Tanca. A structured approach for the definition of the semantics of active databases. *ACM Transaction on Database Systems*, 20(4):414–471, December 1995.

[2] E. Freeman, S. Hupfer, and K. Arnold. *JavaSpaces: Principles, Patterns, and Practice*. The Jini Technology Series. Addison-Wesley, 1999.

[3] D. Gelernter. Generative communication in Linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112, January 1985.

[4] Sun Mycrosystems. Enterprise Java Beans download & specifications, 2000. `http://www.java.sun.com`.

[5] M. Viroli, G. Moro, and A. Omicini. On Observation as a coordination paradigm: an ontology and a formal framework. In *ACM Symposium on Applied Computing – Proceedings of 16th International Conference (SAC01)*, pages 166–175, Las Vegas (NV), USA, March 2001. ACM.

[6] M. Woolridge. *Reasoning about Rational Agents*. The MIT Press, Cambridge, Massachusetts, and London, England, 2000.