# AmI Systems as Agent-Based Mirror Worlds: Bridging Humans and Agents through Stigmergy

Cristiano CASTELFRANCHI [a,1], Michele PIUNTI [b] and Alessandro RICCI [c] and Luca TUMMOLINI [a]

[a] *Istituto di Scienze e Tecnologie della Cognizione, CNR, Via San Martino della Battaglia 44, 00185, Roma, Italy*

[b] *Reply Whitehall, Roma, Italy*

[c] *Alma Mater Studiorum, Università degli studi di Bologna, DEIS, Cesena, Italy.*

**Abstract.**

In this chapter we introduce a vision of agent-oriented AmI systems that is extended to integrate ideas inspired by Mirror Worlds as introduced by Gelernter at the beginning of the 80ies. In this view, AmI systems are actually a digital world mirroring but also augmenting the physical world with capabilities, services and functionalities. We then discuss the value of *stigmergy* as background reference conceptual framework to define and understand interactions occurring between the physical environments and its digital agent-based extension. The digital world augments the physical world so that traces left by humans acting in the physical world are represented in the digital one in order to be perceived by software agents living there and, viceversa, actions taken by software agents in the mirror can have an effect on the connected physical counterpart.

**Keywords.** Mirror worlds, Stigmergy, Cognitive Agents, Ambient Intelligence

## From AmI Systems To Agent-Based Mirror Worlds

As remarked in [Sadri, 2011], Ambient Intelligence (AmI) today can be framed as the convergence of three main areas of computing: ubiquitous computing, sensor network technology and artificial intelligence. Following the ubiquitous computing vision [Weiser, 1993], AmI environments are characterized by the pervasive use of information processing devices thoroughly fused into "the fabric of everyday life until they are undistinguishable from it" [Weiser, 1999], and integrated with other key enabling technologies such as sensors and wireless networks. Then, on this fabric, the software layer exploits Artificial Intelligence techniques along with proper software architectures and paradigm – such as multi-agent systems – to create environments that are sensitive and responsive to inhabitants' needs and capable of anticipating their needs and behavior as well.

[1]Corresponding Author: Istituto di Scienze e Tecnologie della Cognizione, CNR, Via San Martino della Battaglia 44, 00185, Roma, Italy, Email: cristiano.castelfranchi@istc.cnr.it

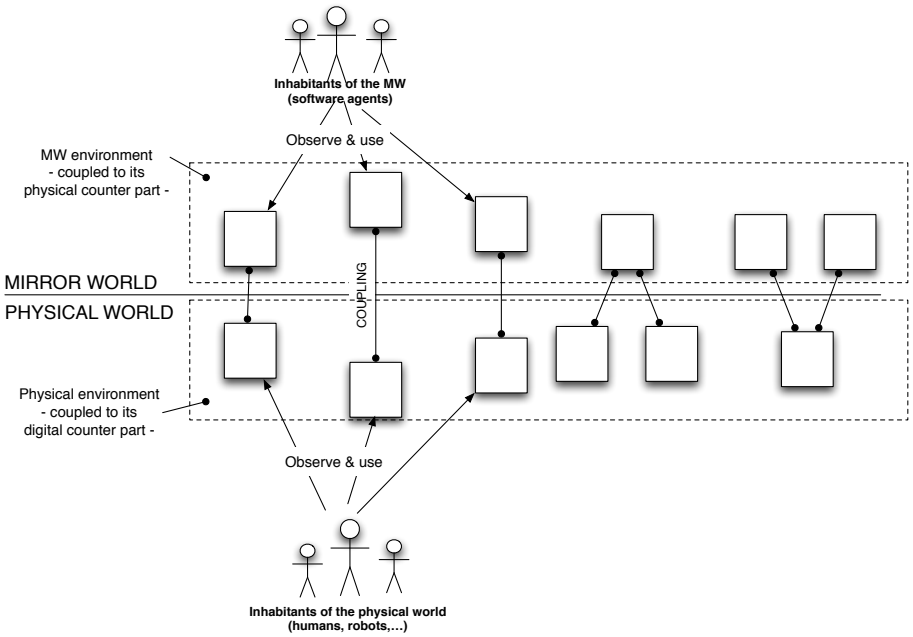


**Figure 1.** A Vision of Agent-Based AmI Systems Integrating Mirror Worlds.

If we consider the AmI literature so far, most of the focus has been on small and quite closed environments, such as smart rooms, smart homes and buildings. The same vision can be extended however also to large, distributed environments, such as full cities. In that case, an inspiring and suggestive view comes from *Mirror Worlds* (MW) idea, introduced in 1982 by David Gelernter [Gelernter, 1992]—the inventor of the tuple spaces coordination model and Linda coordination language [Gelernter, 1985]. In Gelernter's view, mirror worlds are software models of some chunk of reality, "some pieces of the real world going on outside your windows", endlessly poured by oceans of information through hardware and software pipes [Gelernter, 1992]. Using Gelernter's words, they represent a true-to-life mirror image trapped inside a computer, which can be then viewed, zoomed, analyzed by citizens living in the real-world with the help of proper *software assistant agents*. They are meant to be like scientific viewing tools – like microscopes, telescopes – focussed not on hugely large or small items, but on the human-scale social world of organizations, institutions and machines. The final objective is to strongly impact on the life of the citizens of the real-world, who can exploit such tools to tackle the increasing perilous complexity of their government, business, transportation, health, school, university and legal systems.

Putting together these views, we consider here an agent-oriented vision of AmI systems extended towards MW, in which the AmI system is actually a digital world *mirroring* but also *augmenting* the physical world with capabilities, services and functionalities (see Figure 1). The AmI system becomes in this case a kind of a *digital shadow* of the physical world extending it with an open computational layer, strongly coupled with the physical one, structured and organized as an open digital city whose inhabitants are software agents. As in the case of classic AmI system, the bridge between the two layers – the physical and the digital ones – is given by a multitude of heterogeneous networked

(invisible or not) devices, sensors and actuators, making it possible to keep a continuous and consistent coupling between the two layers. Any object of the physical world could have – either explicitly or implicitly – a digital / computational extension in the mirror world representing the object itself, either in terms of a software agent or as part of the agents' environment. Such a digital extension may be possibly perceived also by inhabitants of the physical world through augmented reality-like (or mobile augmented reality) systems. The digital shadow of a physical object would be useful to both enrich or complete its functionalities, and also to actually make the object accessible by the other agents living in the mirror world, so as to perceive and act upon it or interact with it.

To govern complexity and enforce some social order [Castelfranchi, 2000], such agent-based mirror worlds would have explicit organizational structures, normative systems, and related social structures coupled in some way to the organizational and social structures that are defined in the physical human world.

In order to put forward this vision of AmI systems as Agent-based Mirror Worlds, in Section 1 we first suggest that *stigmergy* can be taken as a suitable conceptual framework to understand and build the coupling between the physical and the mirror layers. Stigmergic interaction exploits the power of traces left in a shared environment to support indirect coordination between multiple agents. This mechanism can thus be used also in the case of AmI systems where human and software agents or software agents themselves inhabiting the Mirror World need to coordinate their activities. Then in Section 2, we describe a conceptual and a computational framework that can be used to implement this mechanism with cognitive software agents and discuss some illustrative examples. Finally, in Section 3 we show how some AmI scenarios can be developed on top of this approach.

## 1. Stigmergy to Bridge the Worlds

Stigmergy is a powerful coordination mechanism that has been first explored by the French entomologist Pierre-Paul Grassé to explain the behavior of termites during nest-costruction [Grassé, 1959]. The basic intuition of Grassé has been that traces of "work" left in the environment might become significant stimuli by themselves for other agents and that reliance on such cues is behind the accomplishment of complex collective behaviors. The importance of this mechanism to support indirect coordination has been investigated both in biological [Wenzel, 1991,Karsai and Theraulaz, 1995, Camazine et al., 2001] and artificial societies [Parunak, 1997,Bonabeau et al., 1998, Parunak et al., 2005], often with the aim of showing that even very simple insect-like agents can achieve complex collective behaviors while lacking any cognitive complexity or any knowledge of what they are collectively doing. For instance, it has been suggested that the emergence of pillars, walls and royal chambers in termite nests can be accounted for thanks to the *self-organizing properties* of stigmergy [Boneabeau et al., 1997]. By leaving pheromones in their environment, termites create odor trails that attract other termites. The existence of an initial deposit of soil pellets impregnated with pheromone stimulates workers to accumulate more material. Since termites' behavior follows a simple pattern (i.e. picking up and depositing a soil pellet, if pheromone is present), over time this pattern originates a positive feedback mechanism in which the accumulation of material reinforces the attractivity of deposits through the pheromones emitted by those

materials. Even in absence of any explicit coding of this process in the individual agents, a complex structure that requires collective coordination at the global level can emerge from interactions among its low level components [Garnier et al., 2007]. Inspired by such findings, early approaches to computational stigmergy were aimed at establishing simple mechanisms to promote self regulation processes between multiple entities and have originated the trend of pheromone managing infrastructures [Gambardella et al., 2002, Holland and Melhuis, 1999,Brueckner and Parunak, 2004,Mamei and Zambonelli, 2005].

However, there is no reason why the same mechanism cannot be exploited as well by cognitive agents with more sophisticated abilities. Stigmergic processes with very simple agents actually rely on a rigid and stable behavioral repertoire. On the contrary, trace-based interaction and communication with cognitive agents can benefit of the flexibility of behavior that is enabled by understanding and reasoning about the goal-directed structure of intentional action [Tummolini and Castelfranchi, 2007, Castelfranchi et al., 2010].

With the aim of disentangling the notion of stigmergy from the self-organizing processes that this mechanism can support and to facilitate its application to human interaction, we have thus generalized Grassé's definition and we have proposed that stigmergy occurs whenever another agent's behavioral *trace* (a persistent effect of a practical behavior) is used as a guide for future behavior [Tummolini et al., 2009]. Here it is important to stress that the relevant behaviors that originate stigmergic traces are those that are not specialized only to influence another agent (like verbal language or codified gestures) but those that maintain their original practical function. With this definition in mind, we have thus distinguished two basic stigmergic processes: *stigmergic self-adjustment* and *stigmergic communication*.

There is a case of stigmergic self-adjustment whenever an agent unilaterally exploits the effects of other agents' practical behaviors registered in the environment to avoid possible obstacles or to exploit opportunities. This process creates the pre-requisites for simple forms of indirect coordination. For instance, if an agent leaves his coat on a seat, another one can adjust his behavior accordingly and choose a different place where to sit.

On the other hand, once the agents are able to use each other's traces to coordinate, the traces themselves can be left in the environment on purpose, that is, *in order to* influence those behaviors in some definite manner. That is, the coat can be left on the seat also because one knows that other passengers will understand something from this trace. Stigmergic communication is thus a form of communication which does not exploit any shared code between the agents but only the natural meaning of behaviors. For example, [Clark, 2005] has effectively shown that everyday human joint activities (like that of assembling a TV stand from its parts) rely extensively on *material signals*, that is, on stigmergic communication achieved by deploying material objects, locations or actions around them. For instance, by holding a side piece in front of a co-worker, one can communicate where a top piece must be attached, and coordination in this simple joint action requires that the two agents are able to understand what each is doing and reason accordingly. More generally, once the agents are able to detect the long term traces of the behaviors of each other, they can begin exploiting them in order to improve coordination.

In what follows we suggest that the same kind of stigmergic interaction can be exploited to ensure the coupling between the physical and the mirror layers in the agent-based mirror worlds. In fact, by acting in the extended AmI environment, inhabitants

leave behavioral *traces* that are explicitly reified in some way in the digital extension, so as to be eventually observed and processed by interested inhabitants of the digital world, i.e. software agents. Viceversa, actions in the digital extension done by agents may have eventually effects that are physically perceived by humans, either because of augmented reality systems or because of resulting in changes to the physical environment. So, activities are dynamically co-constructed by joint work of human inhabitants and agents living in the digital layers without an explicit communication, but more in a stigmergic way.

In all this, the notion of *environment* on the agent side becomes a key concept to enable stigmergic interaction, as the medium that reifies the traces left by humans in the physical world and makes them perceivable by the software agents, possibly after doing some processing which is functional to trigger and help interaction and coordination activities.

## 2. How to Implement it? An Approach Based on BDI agents and Artifact-Based Environments

The importance of the environment as a *first-class abstraction* when designing and engineering multi-agent systems [Weyns et al., 2007], as a suitable place where to encapsulate functionalities and services to support agent activities, has been extensively described in the literature [Weyns and Parunak, 2007,Weyns et al., 2005,Ricci et al., 2011]. In this latter view, the environment is no longer just the target of agent's actions and the container and generator of agent percepts as in the traditional AI perspective [Russell and Norvig, 2003], but a part of the MAS that can be suitably designed in order to improve the overall development of the system. The responsibilities and functionalities of environments in this case can be summarised by the following three different levels of support, identified in [Weyns et al., 2007]: *(i) a basic level*, where the environment is exploited to simply enable agents to access the *deployment context*, i.e. the given external hardware/software resources which the MAS interacts with (sensors and actuators, a printer, a network, a database, a Web service, etc.); *(ii) abstraction level*, exploiting an environment abstraction layer to bridge the conceptual gap between the agent abstraction and low level details of the deployment context, hiding such low level aspects to the agent programmer; *(iii) interaction-mediation level*, where the environment is exploited to both regulate the access to shared resources, and mediate the interaction between agents and then enable and support agent coordination. This is the case, e.g., of environment supporting forms of stigmergic coordination in multi-agent systems [Hadeli et al., 2004,Platon et al., 2007,Parunak, 2005,Ricci et al., 2007a]. These levels represent different degrees of functionality that agents can use to achieve their goals.

Among the various approaches, the Agents and Artifacts (A&A) conceptual framework [Omicini et al., 2008,Ricci et al., 2011] introduces a model of environments in Multi-Agent Systems based on *artifacts* as basic first-class abstraction to modularize and structure environment functionalities, especially devised to work within the context of intelligent / cognitive agents.

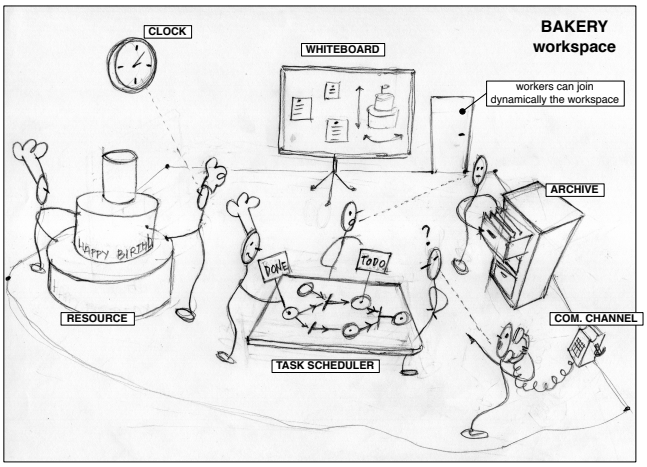


**Figure 2.** Abstract representation of the A&A metaphor in the context of a bakery.

### *2.1. The Agents & Artifacts Meta-model*

By drawing inspiration from Activity Theory [Nardi, 1996], the notion of artifact in MAS has been introduced the first time in [Ricci et al., 2003] in the context of MAS coordination, in particular to define the basic properties of first-class coordination abstractions enabling and managing agent interaction, generalising the notion of coordination media [Omicini et al., 2004]. The concept has been then generalised besides the coordination domain, leading to the definition of the A&A conceptual framework and meta-model and the development of a computational framework – CArtAgO [Ricci et al., 2007b, Ricci et al., 2009b,Ricci et al., 2011] – to support the development and execution of environments designed and programmed upon the notion of artifact.

The background inspiration brought by Activity Theory concerns the role of artifacts in human (as cognitive agents) organizations and working environments. Figure 2 shows a fictional bakery as a toy example. It is a system where articulated concurrent and coordinated activities take place, distributed in time and space, by people working inside a common environment. Mediation tools, artifacts, resources (e.g. a message blackboard, a clock, the task scheduler) are available in the environment in order to ease task fulfillment to entities able to exploit them. Activities are explicitly addressed at *cognitive agents*, individuals able to reason and act in terms of objectives (goals) being informed by data which is available and readable through perception (beliefs). *Interaction* is a main dimension, due to the dependencies among the activities. As well as cooperation, interaction is enabled and strongly promoted either by means of message based communication and through *environment* infrastructures explicitly engineered for supporting it. So the environment – as the set of tools and resources used by people to work – plays a key role in performing tasks efficiently. Besides tools, the environment hosts resources that represent the co-constructed results of people work (e.g. the cake). The complexity of work recalls for some division of labor and decentralized, *distributed work spaces*, so each person is responsible for the fulfillment of a series of situated tasks.

A&A brings this idea to multi-agent systems, so that a MAS environment is designed and programmed in terms of a dynamic set of artifacts as first-class computational enti-

ties, collected in localities called *workspaces*. Artifacts represent resources and tools that agents can dynamically instantiate, share and use to support their individual and collective activities [Omicini et al., 2008]. On the one side, they are first-class abstractions for MAS designers and programmers, who define the types of artifacts that can be instantiated in a specific workspace, defining their structure and computational behavior. On the other side, artifacts are first-class entities of agents world, which agents perceive, use, compose, and manipulate as such.

To make its functionalities available and exploitable by agents, an artifact provides a set of operations and a set of observable properties. Operations represent computational processes – possibly long-term – executed inside artifacts, and finally correspond to the *actions* that agents have to act upon the artifact. The term *usage interface* is used to indicate the overall set of artifact operations available to agents. Observable properties represent state variables whose value can be perceived by agents[2]; the value of an observable property can change dynamically, as result of operation execution. The execution of an operation can generate also *signals*, to be perceived by agents as well: differently from observable properties, signals are useful to represent non-persistent observable *events* occurred inside the artifact, carrying some kind of information. Besides the observable state, artifacts can have also an hidden state, which can be necessary to implement artifact functionalities.

A straightforward example of artifacts in A&A systems recalling the human world is represented by digital agendas. Agenda artifacts can indeed be defined cognitive artifacts, namely resources which can be exploited by agents in complex systems in order to externalize, and share, the schedule of their cooperative tasks/activities. Agendas are thus designed for being *cognitively* exploited, created and even shared by societies of software agents. Their main functionality is to generate a signal (alarm) at the scheduled time, so as to allow observer agents to react accordingly and fulfill the programmed task. Accordingly, agendas are designed to make it observable their relevant properties, namely the schedule list in this case. In so doing, agendas have the additional function to inform agents, allowing them to read and become aware about the state of their next tasks.

### *2.2. BDI Agents Happily Living in Artifact-Based Environment*

Even if orthogonal with respect to agent dimension [Ricci et al., 2008], the basic model for artifacts has been specifically conceived to suite high-level models and architectures for agents, such as the Belief-Desire-Intention (BDI) one [Ricci et al., 2011].

In particular, by adopting an artifact-based perspective, an operation provided by an artifact *is* an external action[3] available to every agent working in the same workspace where the artifact is. So the repertoire of external actions available to an agent is defined by the set of artifacts that populate the environment. This implies that the actions repertoire can be dynamic, since the set of artifacts can be changed dynamically by agents themselves, instantiating new artifacts or disposing existing artifacts. In this perspective, artifacts can be framed as tools *externalizing* agent capabilities [Ricci et al., 2009a], ex-

[2] Actually by those agents that are observing the artifact, as will be clarified later on in the section.

[3] By adopting a terminology typically used in agent-oriented programming languages, especially BDI ones, external actions are meant to have an effect on the environment, internal actions instead on agent internal state.

tending - in a sense - their minds, like in the Extended Mind perspective suggested by Clark in cognitive science [Clark and Chalmers, 1998].

Observable properties and events constitute instead agent percepts. In BDI architectures – as implemented in particular in agent programming languages and platforms such as Jason [Bordini et al., 2007] – percepts related to the value of observable properties can be directly modelled inside agents as beliefs about the actual state of the environment. Actually, to scale up with the environment complexity, in artifact-based environments an agent can dynamically select which are the artifacts to observe, so as to perceive the observable properties and events of only that part of the environment that the agent is interested in. So, the set of beliefs of a BDI agent working inside an artifact-based environment is given by the set of observable properties of all the artifacts that the agent decided to observe.

To explore concretely these ideas and to enable the development of cognitive agent applications with artifact-based environments, CArtAgO – which is the reference computational framework and infrastructure implementing the A&A model – has been integrated with different BDI agent programming languages and frameworks. JaCaMo[4] [Boisser et al., 2012] in particular is a comprehensive platform, which allows for developing and running multi-agent systems composed by BDI agents programmed in Jason working inside CArtAgO distributed artifact-based environment, organised into organisations specified in $\mathcal{M}$OISE.

### *2.3. Stigmergy in Artifact-Based Environments*

By adopting the A&A perspective, stigmergic processes can be designed and implemented in CArtAgO by introducing suitable artifacts to support agents' work. In spite of the peculiarities of the specific process, it is possible here to identify some basic features that characterise these artifacts, defining a sort of abstract architecture that can be specialised according to the specific context. This abstract architecture will be applied in the social case study described in the next section, in particular to drive the design and implementation of artifacts supporting the stigmergic coordination of different teams of cognitive agents.

#### *2.3.1. The Simplest Scenario: Two Agents*

To describe these features, we start from considering here the simplest scenario involving stigmergy, composed by an agent $A$ performing some task $G_1$, whose behavioral traces are used as a guide—i.e. as *cue* [Tummolini et al., 2009]—by another agent $B$ to perform some task $G_2$.

In order to model this basic scenario, we introduce an artifact $T$ mediating the work of the agent $A$ with respect to its task, that is an instrument that would factorise and encapsulate some basic functionalities useful to achieve the objective of the work (see Figure 3, left). On the one hand, from the agent point of view, the artifact represents the means that helps or is necessary to *enact* his practical behavior: for this purpose, artifact $T$ is designed with a suitable usage interface accessed by $A$ to do its work. On the other hand, from the designer of the stigmergic process, the artifact functions as the means to keep track and elaborate agent practical behavior, by creating and recording in its internal

[4] http://jacamo.sourceforge.net

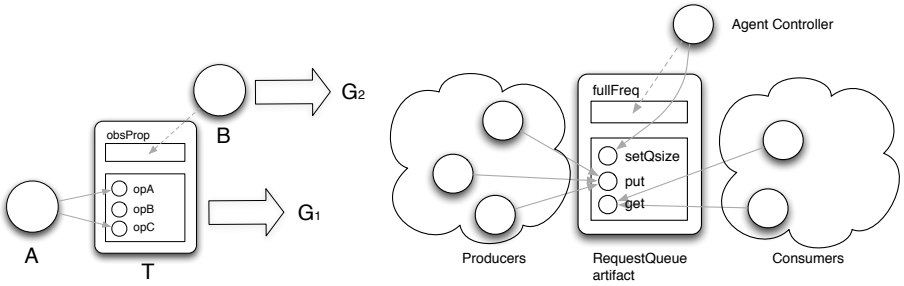


**Figure 3.** *(left)* The artifact T makes it observable to agent B the traces of agent A's practical work; *(right)* The stigmergic producers-consumers example

state the *traces* of agent work. The traces can be simply logs of agent behavior, or, in the more general case, the result of some kind of pre-computations which are functional to the stigmergic processes, such as *aggregation*, *ordering*, *selection*.

Then, the artifact $T$ must be designed so as to make those traces observable by agent $B$: this is simply realised by making the traces *observable properties* of the artifact, and let the agent observe the artifact $T$—in order to have such information among its beliefs (in the case of BDI architecture).

Finally, the *relevance* of traces changes not only with agent interaction but also with time passing: for this purpose, the artifact $T$ can embed some basic time-driven operations that changes the traces so as to module their persistence in time.

A variant to this basic schema useful for fully distributed contexts, is to consider $A$ in a workspace $W_1$ and $B$ being situated in a different workspace $W_2$. In that case, we split the functionality of the artifact $T$ in two artifacts $T_1$ and $T_2$, linked together: $T_1$ used by $A$ in $W_1$, functioning as mediator of $A$'s activities; $T_2$ observed by $B$ in $W_2$, with observable properties representing the trace; the pre-computation of the trace can be done either by $T_1$ or $T_2$, according to the need.

### *2.3.2. The Stigmergic Producers-Consumers Example*

The basic two-agents scenario can be generalised by considering—instead of agents $A$ and $B$—an agent society composed of agent playing different roles, interacting in order to perform some global tasks, some of them using the traces of the work of others to enact the work that may influence back then the work of the formers.

As a more concrete but still simple example, let's consider a MAS used in a service-oriented scenario, with $A_1$ as a dynamic set of agents acting as requesters of a service $S$ (i.e. request producers), $A_2$ as a dynamic set of agents acting as providers of service $S$ (i.e. request consumers). A `RequestQueue` artifact is used as a bounded queue (buffer) to uncouple and coordinate producers-consumers interaction (see Figure 3, right). The usage interface of the artifact includes a `put` operation control to insert a new request—exploited by $A_1$ agents—and a `get` operation control to remove the first available request.

Then, we want to introduce in the system an agent $B$ who is in charge of optimizing the overall producers-consumers society behavior, in particular by taking some kind of actions—e.g. to augment the number of service providers—in the case of critical situations—e.g. `RequestQueue` is full with a frequency in time greater than some threshold $Th$. For doing this, we enhance the `RequestQueue` artifact with an observ-

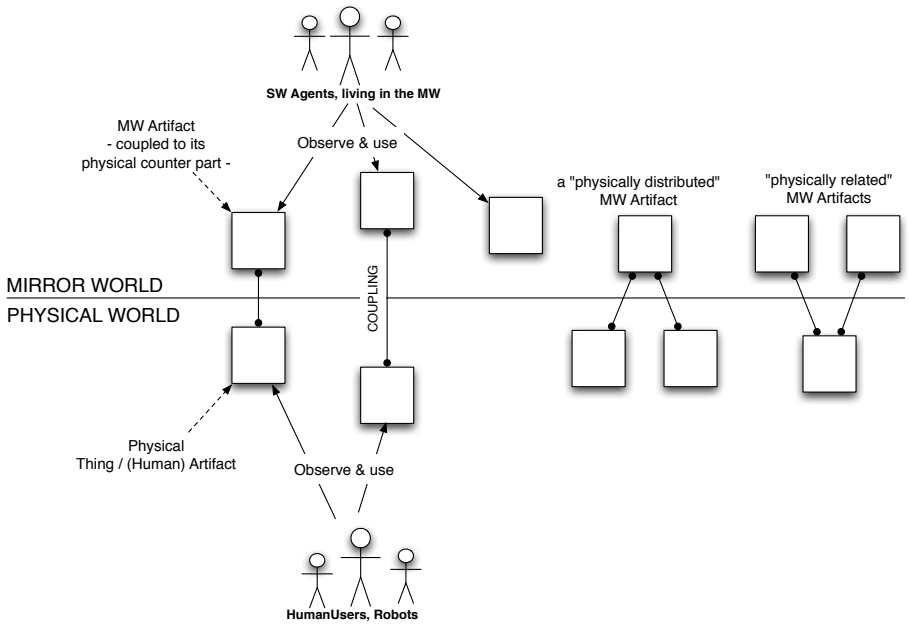


**Figure 4.** Artifact-Based Mirror Worlds coupled to the Physical World

able property `fullFreq`, containing the percentage of time the bounded queue has been full—let's say—in the last hour, and the necessary inner working machinery to compute and make such value available. The value of `fullFreq` can be considered a trace of producers-consumers work in the overall, and relevant in this case for the work of the agent controller $B$, observing `RequestQueue`.

### *2.3.3. Stigmergic Self-Adjustment*

Recalling back the two-agents examples, in many cases the task $G_2$ of agent $B$ includes influencing back the work of agent $A$, by improving—for instance—its performance, taking into the account contingencies that are not known or are beyond agent $A$'s competences. For this purpose, artifact $T$ may expose further operations to be exploited not by agent $A$ but by agent $B$, which are functional to adapt artifact behavior or functionalities in order to improve the overall performance of the subsystem $A$ plus $T$.

By referring to the producers-consumers example, the `RequestQueue` artifact can be equipped with an operation `setQSize` that makes it possible to dynamically change the *size* of the queue. This operation is meant to be exploited by agent controller $B$ applying some policy relating the course of `fullFreq` value and the size of the queue, for instance deciding to augment the queue size as soon as `fullFreq` exceeds some threshold $Th$.

## 3. Putting the Pieces Together: Designing Mirror Worlds with BDI Agents, Artifacts and Stigmergy

The A&A conceptual framework and related specific technologies such as CArtAgO and JaCaMo make it possible to exploit agents and artifacts as basic bricks to conceive mir-

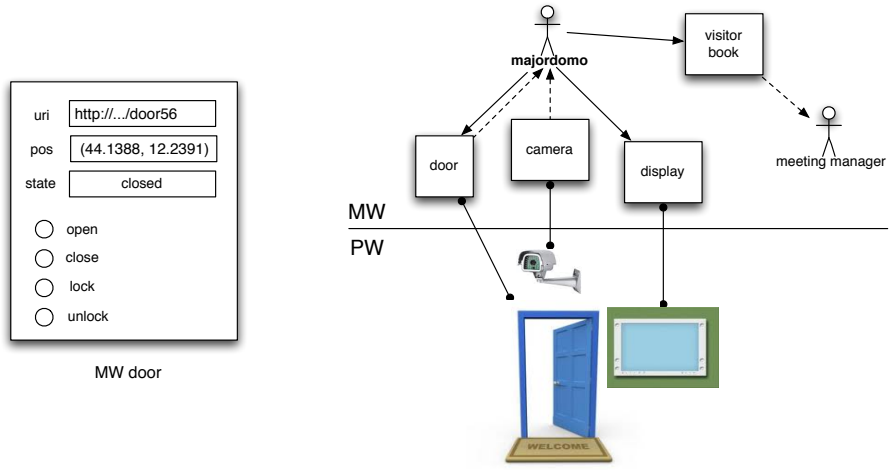


**Figure 5.** A MW door artifact and a simple usage scenario

ror worlds as depicted in the Introduction, using stigmergy both to frame the conceptual coupling between the physical and digital layers and to conceive the coordination inside cooperative activities that are co-constructed by inhabitants of such layers, i.e. humans and software agents.

In particular, cognitive software agents are the autonomous citizens of the mirror world and artifacts are used as basic bricks to shape the mirror world environment, so that: *(a)* the agents act and perceive in a distributed computational environments which is the key element to create a strong coupling – temporal and spatial – between the two layers, the physical one and the virtual one; *(b)* the agent computational environment is properly designed and instrumented to enable stigmergic interaction between human and cognitive agents, and between agents themselves.

So in a mirror world, some artifacts can be used to directly represent the *digital* counterpart of things or artifacts belonging to the physical world, which can be observed and affected by agents of the MW (see Figure 4). Such MW artifacts are meant to be strongly coupled to their physical extension by means of proper low-level coupling technologies, so as to provide some level of consistency between their (observable) states. Actually the specific shape of a MW artifact – observable properties, operations – coupled to a physical artifact can vary depending on the functionalities that we want to provide and the available coupling technologies. So, from a MW developer's viewpoint, these kinds of artifacts are a way to engineer the bidirectional information flow between a piece of physical environment and its counter-part in the mirror. In fact, the MW artifacts not only provide a way to represent the physical one (and make it observable) at the MW level, but also encapsulate some specific processing and elaboration, concerning artifact functionalities, and provide an action interface to possibly affect the physical side.

As a toy example, let's take a door (Figure 5, on the left). A possible MW door artifact coupled to the physical door could have an observable property the state (open, closed, locked) and some operations (actions) to change the state (open, close, lock). As soon as the physical door – being closed – is opened or locked (by humans), the observable property of the MW artifact is updated accordingly. On the other side, as soon

as an agent in the MW will perform a *lock* action on the MW door, the door would be locked also in the physical world. A very simple scenario including the MW door artifact is shown in Figure 5 (on the right), representing a possible fragment of MW ruling the access of people to some room/building and including also a MW camera and a MW display artifacts, coupled with a physical camera and display, located near the door, and a visitor book, an artifact uncoupled from the physical world used to keep track of visitors that are inside the place. A *majordomo* agent is responsible to regulate the access and welcome visitors according to some strategy, by observing information provided by the camera and the door, and acting on the door itself and the display (to show messages), as well as on the visitor book. Actually the same MW artifacts could be shared and used by other agents – not known to majordomo – to do their jobs inside the MW.

Actually the type or granularity of such MW artifacts is a matter of design and could depend on the specific application. We could have – for instance – a MW artifact representing a whole room (abstracting from devices inside), as well as even raw sensors or actuators – such as a camera, a thermometer or a switch. In any case, MW artifacts coupled with physical things all have some basic common properties, which include the identifier of the physical resource and its geo-spatial information.

As a further, more articulated, example, suppose to have a MW running on top of a University campus, to ease the academic life of students and teachers. Every teacher as well as every student has a smart-phone, running her personal assistant agents, situated in the University MW. The MW includes:

- some *room* artifacts coupled with rooms where the lectures take place;
- a *course portal* artifact for each course – showing, among the other things, information about the next lecture—the room where it will take place, when, its state (stated, finished, cancelled).

The idea is to exploit actions that the teachers do on the physical environment to automatically trigger activities at the MW levels and automate tasks related to the coordination with students and other personal staff of the university.

When there is not a lecture running, a lecture room is locked. A lecture room can be unlocked then by a teacher who is going to have a lecture in the room. The action of unlocking and entering the room by a teacher at a time slot corresponding to her lecture can be understood – at the MW level – as the fact that the lecture is going to start: so, as soon as a teacher unlocks and enters the room, her personal agent – who is aware of the course agenda and tracking the room state – acts on the MW course portal artifact to change the state of the next (current) lecture. Such a change can be perceived then by the personal agents of the course students, who may want to notify the event to the interested student (by a message on the the smartphone) in the case that she is not already inside the room.

Then, suppose that there were a problem in the room that require to do the lecture elsewhere—e.g., the projector is broken. The teacher can notify the problem and then select another available room, assisted by her personal agent acting on the MW room artifact and other artifacts of the MW campus. Moreover, the MW room artifact provides an action to the teacher's personal agent to post a message reporting that there will be a lecture in a while on the physical display of the new room and that the lecture has been moved on the physical display of the current one. The agent can also to notify the change on the MW course portal artifact, so that those students who aren't arrived yet can be

informed of the change by their personal agent. Finally, the notification of the problem on the MW room artifact triggers the reaction of a building maintenance agent – who is observing all the rooms of the campus and is thus in charge of taking some action.

Besides these simple examples, the approach conceptually scales well considering large, open and complex environments, from quarters and districts to full cities. In that case artifacts and agents embodying such mirror worlds would be necessarily organized in terms of multiple workspaces distributed over the network, eventually running on cloud services. The capability of dynamically creating and disposing artifacts (by the agents themselves), as well as to define organizational structures and rules governing the interactions inside the agent-based digital layer is essential to fully handle the dynamism and openness which is implied by such complex environments. To this purpose, as an example, an agent platform and infrastructure like JaCaMo provides an explicit support to define the organization of the overall multi-agent systems, based on the $\mathcal{M}$OISE organizational model, fully integrated with the agent and environment dimensions.

## 4. Conclusion

It is not hard to preview that the design and engineering of large, fully distributed and open Ambient Intelligence systems – scaling from rooms to full cities and beyond – will be a main point in the agenda of AmI research in the short future. In that perspective, besides the continuous development of enabling HW technologies and AI techniques, and the adoption of standards for ensuring interoperability, an important issue concerns the availability of proper conceptual frameworks and software abstractions to tackle the inherent complexity in modeling and designing such systems. In this contribution we envisioned an agent-oriented vision of Ambient Intelligence systems integrating ideas from Gelernter's mirror worlds and existing models and technologies for multi-agent oriented programming, aiming at providing a uniform conceptual framework to conceive and develop such complex systems. In that view, we discussed the role of *stigmergy* as a natural interaction model to create the communication and coordination between the physical and digital layers, i.e. between the human inhabitants of the physical word and the software agents living in the digital shadow of that world.

## References


[Boisser et al., 2012] Boisser, O., Bordini, R., Hubner, J., and Ricci, A. (2012). Multi-agent oriented programming with jacamo. *Science of Computer Programming*. to appear.

[Bonabeau et al., 1998] Bonabeau, E., Henaux, F., Guérin, S., Snyers, D., Kuntz, P., and Theraulaz, G. (1998). Routing in telecommunications networks with ant-like agents. In *IATA '98: Proceedings of the second international workshop on Intelligent agents for telecommunication applications*, pages 60–71, London, UK. Springer-Verlag.

[Boneabeau et al., 1997] Boneabeau, E., Theraulaz, G., Deneubourg, J., Franks, N., Rafelsberger, O., Joly, J., and Blaco, S. (1997). A model for the emergence of pillars, walls and royal chambers in termite nests. *Philosophical Transactions of the Royal Society of London*, 353:1561–1576.

[Bordini et al., 2007] Bordini, R. H., Hübner, J. F., and Wooldrige, M. (2007). *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley Series in Agent Technology. John Wiley & Sons.

[Brueckner and Parunak, 2004] Brueckner, S. A. and Parunak, H. V. D. (2004). Self-organizing MANET management. In Di Marzo Serugendo, G., Karageorgos, A., Rana, O. F., and Zambonelli, F., editors, *Engineering Self-Organising Systems: Nature-Inspired Approaches to Software Engineering*, volume 2977 of *LNAI*, pages 20–35. Springer.

[Camazine et al., 2001] Camazine, S., Deneubourg, J., Franks, N., Sneyd, J., Theraulaz, G., and Bonabeau, E. (2001). *Self-Organization in Biological Systems*. Princeton University Press, Princeton, NJ.

[Castelfranchi, 2000] Castelfranchi, C. (2000). Engineering social order. In *Proceedings of the First International Workshop on Engineering Societies in the Agent World: Revised Papers*, ESAW '00, pages 1–18, London, UK. Springer-Verlag.

[Castelfranchi et al., 2010] Castelfranchi, C., Pezzulo, G., and Tummolini, L. (2010). Behavioral implicit communication (bic): Communicating with smart environments via our practical behavior and its traces. *International Journal of Ambient Computing and Intelligence*, 2(1):1–12.

[Clark and Chalmers, 1998] Clark, A. and Chalmers, D. (1998). The extended mind. *Analysis*, 58: 1:7–19.

[Clark, 2005] Clark, H. (2005). Coordinating with each other in a material world. *Discourse Studies*, 7:507–525.

[Gambardella et al., 2002] Gambardella, L. M., Dorigo, M., Middendorf, M., and Stützle, T. (2002). Special section on ant colony optimization. *IEEE Transactions on Evolutionary Computation*, 6(4):317–365.

[Garnier et al., 2007] Garnier, S., Gautrais, J., and Theraulaz, G. (2007). The biological principles of swarm intelligence. *Swarm Intelligence*, 1(1):3–31.

[Gelernter, 1985] Gelernter, D. (1985). Generative communication in linda. *ACM Trans. Program. Lang. Syst.*, 7:80–112.

[Gelernter, 1992] Gelernter, D. H. (1992). *Mirror Worlds: or the Day Software Puts the Universe in a Shoebox...How It Will Happen and What It Will Mean*. Oxford.

[Grassé, 1959] Grassé, P. (1959). La reconstruction du nid et les coordinations inter- individuelles chez bellicositermes natalensis et cubitermes. La théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux*, 6:41–81.

[Hadeli et al., 2004] Hadeli, K., Valckenaers, P., Kollingbaum, M., and Van Brussel, H. (2004). Multi-agent coordination and control using stigmergy. *Comput. Ind.*, 53:75–96.

[Holland and Melhuis, 1999] Holland, O. and Melhuis, C. (1999). Stigmergy, self-organization, and sorting in collective robotics. *Artificial Life*, 5(2):173–202.

[Karsai and Theraulaz, 1995] Karsai, I. and Theraulaz, G. (1995). Nest building in a social wasp: postures and constraints. *Sociobiology*, 26:83–114.

[Mamei and Zambonelli, 2005] Mamei, M. and Zambonelli, F. (2005). Programming stigmergic coordination with the tota middleware. In *4th ACM International Joint Conference on Autonomous Agents and Multiagent Systems*, New York, USA. ACM.

[Nardi, 1996] Nardi, B. A. (1996). *Context and Consciousness: Activity Theory and Human-Computer Interaction*. MIT Press.

[Omicini et al., 2008] Omicini, A., Ricci, A., and Viroli, M. (2008). Artifacts in the A&A meta-model for multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 17 (3).

[Omicini et al., 2004] Omicini, A., Ricci, A., Viroli, M., Castelfranchi, C., and Tummolini, L. (2004). Coordination artifacts: Environment-based coordination for intelligent agents. In Jennings, N. R., Sierra, C., Sonenberg, L., and Tambe, M., editors, *Proc. of AAMAS'04*, volume 1, pages 286–293, New York, USA. ACM.

[Parunak, 2005] Parunak, H. V. D. (2005). A survey of environments and mechanisms for human-human stigmergy. In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *E4MAS*, volume 3830 of *Lecture Notes in Computer Science*, pages 163–186. Springer.

[Parunak, 1997] Parunak, V. (1997). Go to the ant: Engineering principles from natural agent systems. *Annals of Operations Research*, 75:69–101.

[Parunak et al., 2005] Parunak, V., Brueckner, S., and Sauter, J. (2005). Digital pheromones for coordination of unmanned vehicles. In Weyns, D., Parunak, V., and Michel, F., editors, *Environments for Multiagent Systems I*, number 3374 in Lecture Notes in Computer Science, pages 179–183. Springer-Verlag, Berlin / Heidelberg.

[Platon et al., 2007] Platon, E., Mamei, M., Sabouret, N., Honiden, S., and Parunak, H. V. (2007). Mechanisms for environments in multi-agent systems: Survey and opportunities. *Autonomous Agents and Multi-Agent Systems*, 14:31–47.

[Ricci et al., 2003] Ricci, A., Omicini, A., and Denti, E. (2003). Activity Theory as a framework for MAS coordination. In Petta, P., Tolksdorf, R., and Zambonelli, F., editors, *Engineering Societies in the Agents World III*, volume 2577 of *LNCS*, pages 96–110. Springer. 3rd International Workshop (ESAW 2002), Madrid, Spain, 16–17 September 2002. Revised Papers.

[Ricci et al., 2007a] Ricci, A., Omicini, A., Viroli, M., Gardelli, L., and Oliva, E. (2007a). Cognitive stig-

mergy: Towards a framework based on agents and artifacts. In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for MultiAgent Systems III*, volume 4389 of *LNAI*, pages 124–140. Springer.

[Ricci et al., 2008] Ricci, A., Piunti, M., Acay, L. D., Bordini, R., Hubner, J., and Dastani, M. (2008). Integrating Artifact-Based Environments with Heterogeneous Agent-Programming Platforms. In *Proc. of AAMAS'08*.

[Ricci et al., 2009a] Ricci, A., Piunti, M., and Viroli, M. (2009a). Externalisation and Internalization: A New Perspective on Agent Modularisation in Multi-Agent Systems Programming. In Dastani, M., Seghrouchni, A. E. F., Leite, J., and Torroni, P., editors, *Proceedings of MALLOW 2009 federated workshops: LAnguages, methodologies and Development tools for multi-agent systemS (LADS 2009)*.

[Ricci et al., 2011] Ricci, A., Piunti, M., and Viroli, M. (2011). Environment programming in multi-agent systems: an artifact-based perspective. *Autonomous Agents and Multi-Agent Systems*, 23:158–192. 10.1007/s10458-010-9140-7.

[Ricci et al., 2009b] Ricci, A., Piunti, M., Viroli, M., and Omicini, A. (2009b). Environment programming in CArtAgO. In *Multi-Agent Programming: Languages, Platforms and Applications, Vol. 2*, pages 259–288. Springer.

[Ricci et al., 2007b] Ricci, A., Viroli, M., and Omicini, A. (2007b). CArtAgO: A framework for prototyping artifact-based environments in MAS. In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for MultiAgent Systems III*, volume 4389 of *LNAI*, pages 67–86. Springer.

[Russell and Norvig, 2003] Russell, S. and Norvig, P. (2003). *Artificial Intelligence, A Modern Approach (2nd ed.)*. Prentice Hall.

[Sadri, 2011] Sadri, F. (2011). Ambient intelligence: A survey. *ACM Comput. Surv.*, 43(4):36.

[Tummolini and Castelfranchi, 2007] Tummolini, L. and Castelfranchi, C. (2007). Trace signals: The meanings of stigmergy. In Weyns, D., Parunak, V., and Michel, F., editors, *Environments for Multi-Agent Systems III*, number 4389 in Lecture Notes in Artificial Intelligence, pages 141–156. Springer-Verlag, Berlin / Heidelberg.

[Tummolini et al., 2009] Tummolini, L., Mirolli, M., and Castelfranchi, C. (2009). Stigmergic cues and their uses in coordination: An evolutionary approach. In Uhrmacher, A. and Weyns, D., editors, *Agents, Simulation and Applications*, pages 243–265. CRC Press.

[Weiser, 1993] Weiser, M. (1993). Some computer science issues in ubiquitous computing. *Commun. ACM*, 36:75–84.

[Weiser, 1999] Weiser, M. (1999). The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3:3–11.

[Wenzel, 1991] Wenzel, J. (1991). Evolution of nest architecture. In Ross, K. and Matthews, R., editors, *The social biology of wasps*, pages 480–519. Cornell University Press.

[Weyns et al., 2007] Weyns, D., Omicini, A., and Odell, J. J. (2007). Environment as a first-class abstraction in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):5–30. Special Issue on Environments for Multi-agent Systems.

[Weyns and Parunak, 2007] Weyns, D. and Parunak, H. V. D., editors (2007). *Journal of Autonomous Agents and Multi-Agent Systems. Special Issue: Environment for Multi-Agent Systems*, volume 14 (1). Springer Netherlands.

[Weyns et al., 2005] Weyns, D., Parunak, H. V. D., Michel, F., Holvoet, T., and Ferber, J. (2005). Environments for multiagent systems: State-of-the-art and research challenges. In Weyns, D., Parunak, H. V. D., Michel, F., Holvoet, T., and Ferber, J., editors, *Environment for Multi-Agent Systems*, volume 3374, pages 1–47. Springer-Verlag, Berlin-Heidelberg.