Implicit Culture and Multi-agent Systems

Enrico Blanzieri, Paolo Giorgini, and Fausto Giunchiglia

Abstract— **Agents' autonomy and incoming agents can prevent a multi-agent system from fulfilling its requirements. Given a group of autonomous agents acting in an environment it should be useful to exploit information about their actions in order to improve the single agent's knowledge and behavior, and guarantee the realization of the requirements. In this paper, we introduce the concept of implicit culture and propose a general architecture for Systems for Implicit Culture Support (SICS). We show how it is possible to use a SICS to guarantee the persistence of the multi-agent system's requirements; we present some existing systems that can be considered as instances of our architecture; and finally, we consider the related work.**

Keywords— **Agents, multi-agent systems, implicit culture, behaviors, actions.**

I. Introduction

Different methodologies are used for the requirements modeling of a multi-agent system. For instance, in [1] the authors describe the system in terms of roles and interactions, whereas in [2], [3] the system is modelled in terms of agents and intentional dependencies. However, autonomy of the agents, unknown properties of the environment and insertion of new agents do not allow to foresee completely the system's behavior in the modeling stage. As a consequence, the overall system can fail to fulfill the desired requirements. In particular, requirements should persist after a change in the composition of the group of agents that have to be considered as autonomous and operating in an incompletely known environment.

Let's concentrate on a single agent that joins a group of agents and begins to act in an environment. Its behavior is far from optimal and the other agents would probably have more knowledge and would be more skilled. Moreover, they might not be willing to share their knowledge and in same cases they might not even able to represent or communicate it. The insertion of the new agent can seriously threaten the fulfillment of the requirements.

In order to have the persistence of the requirements the new agent should act consistently with the culture of the group. In fact, in this "new kid in town" scenario the agent is not able to cope with the environment and with the other agents. More depressingly, the group of agents knows this and actively exploits it. In the case of humans this phenomenon is sometimes referred to as "cultural shock". In fact, knowledge about the environment and about the behaviors of the agents is part of their culture and that is what the new agent lacks.

The problem of having the new agent act consistently with the knowledge and behaviors of the group could be solved by improving the capabilities of the agent in terms of communication, knowledge and learning. The first solution is "just ask someone" but, in an agent setting, this is *not* a simple solution. It is necessary to know what to ask (knowledge about the problem), how to ask (a language for expressing the problem), and who to ask to (some brokering facility). More fundamentally, it is also necessary to know that one has a problem in the first place, and to have its solution among the goals. The second possible solution is to represent the relevant knowledge and provide it to the agent. If the knowledge required is objective and relatively static, the representation can be done by observing the environment and describing it. Building ontologies is a common way of addressing this problem. Unfortunately, the environment can be partially unknown and intrinsically dynamic. As a third option, it is possible to equip the agent with both observational and learning capabilities and acquire skills by imitation of the other agents. One drawback is that these capabilities are rather complex and their application requires resources.

Improving the capabilities of the single agent is not an available option if the agent has a high level of autonomy. An autonomous agent performs its actions without direct external control and without external access to its internal states. Moreover, autonomy itself can produce behaviors that were not clearly foreseen during the requirements definition and the design phases of the system. Some of them can be different from the desired ones, for instance overload of communication or task assignments to a subset of agents or inactivity of some others. Other behaviors can be unexpected without incompatibility with the requirements of the system, for instance, a pattern of task assignments can emerge.

The persistence of the requirements is handled by the actions of new agents and by autonomous behavior performed by the group. Not availability of knowledge on the environment and autonomy itself prevent to address directly the problem. Moreover, the autonomy of the agents has produced an unforeseen behavior that would be interesting to exploit in order to fulfill the requirements.

When the environment is partially under control, the problem can be tackled in a very different way. Instead of working on the agent capabilities, it is possible to modify the view that the agent has of the environment and consequently its actions without violating its autonomy. In fact, changing in a proper way the set of possible actions that the agent can perform in the environment can lead the agent to act consistently with the behavior a member of the group. The group itself can optimize its behavior for the particular environment. Moreover, neither the new agent nor a member of the group is required to know this

E. Blanzieri is with ITC-IRST, Via Sommarive 18, Povo, 38050 Trento - Italy. Email:blanzier@itc.it

P. Giorgini is with DISA - University of Trento, Via Inama 5, 38100 Trento - Italy. Email:pgiorgini@cs.unitn.it

F. Giunchiglia is with both ITC-IRST and DISA - University of Trento. Email:fausto@irst.itc.it

and so they share the same culture in an implicit way.

In the present paper we introduce the concept of *implicit culture*, namely the relation between groups of agents that behave according to a cultural schema and groups that contribute to the production of that cultural schema. A critical condition is that the members of the former group have no need to know about the latter, its members or their behavior. We also define *implicit culture phenomenon* as a pair of groups acting in an implicit culture relation. Moreover, we propose an architecture for systems aimed to support the emergence of an implicit culture phenomenon on groups of agents and we show how implicit culture solves the problem of the sub-optimal behavior of a set of new agents. The architecture is very general and covers, as special cases, systems such as Collaborative Filtering [4].

The paper is organized as follows: the next section introduces the concept of implicit culture; section III presents an architecture for supporting it; section IV shows some instances of the architecture; and finally, sections V and VI describe related work and draw conclusions, respectively.

II. Implicit Culture

A group of agents effectively acting in an environment exploits a great amount of knowledge and skills. When new agents are introduced in the environment they face the problem of acquiring the necessary knowledge. The problem of the new agents would be solved if they acted in a way consistent with the knowledge and behaviors of the group. If the environment is under control and modifiable it is possible to obtain the same effect without the need for the agents to know about the group and its behavior. In the following we will informally introduce the notion of implicit culture as a way to capture this phenomenon.

We assume that the agents perceive and act in an environment composed of objects and other agents. In this perspective, agents are objects that are able to perceive, act and, as a consequence of perception, know. A set of agents is a group.

Actions can have as arguments objects, as in *offer(book1,price1)* or *demand(book2,price2)*, agents, as in *look_for(buyer)* or *ask_about(seller)*, or both objects and agents, as in *send(message,seller)*. Before executing an action, an agent faces a scene formed by a portion of the environment, namely objects and agents, and actions that are possible in it (as a special case the agent can be part of the scene when reflexive actions are possible). For example, an agent *buyer* faces *seller1, seller2, book1, gadget1, price1, price2* and can perform *buy_from(seller1,book1,price1)*, *buy_from(seller2,gadget1,price2)* and *buy_nothing()*. Hence, an agent executes an action in a given situation, namely the scene faced by the agent at a given time, so the agent executes situated actions. For example, the agent *buyer* executed the action *buy_from(seller1,book1,price1)* while he was facing the scene composed of *seller2*, *price2* and the possible actions.

After a situated action has been executed the agents face a new scene. At a given time the new scene depends on the environment and on the situated executed action. If *buyer1* performs *buy_from(seller1,antique_book1,price1)*, and *buyer2* performs *do_nothing()*, both *buyer1* and *buyer2* will have the scenes they face changed for *antique_book1* is not on sale anymore. If *seller* performs *sell_to(buyer1,antique_book1,price1)*, the next scene it faces will not include *antique_book1*.

The situated executed action that an agent chooses depends on its private states and in general it is not deterministically predictable with the information available externally. Rather, we assume it can be characterized in terms of probability and expectations. As an example, given a *buyer* facing a scene in which it can perform *buy_from(seller1,book1,unreasonable_price)*, *buy_from(seller2,book1,low_price)* or *buy_nothing()* the expected situated action can be *buy_from(seller2,book1,low_price)*.

Given a group of agents, let us suppose that there exists a theory about their expected situated actions. If the theory is consistent with the executed actions of the group, it can be considered a validated cultural constraint for the group. The theory captures the knowledge and skills of the members about the environment. For instance:

$$\begin{aligned} &\forall x, y \in Group, book \in Books : \\ &\quad execute(x, buy_from(y, book, p)) \wedge \\ &\quad execute(y, sell_to(x, book, p)) \rightarrow \\ &\quad p \in [low(book), high(book)] \end{aligned} \tag{1}$$

expresses that, for all agents of the group and all books, if a buyer buys a book from a seller (and the seller sells the book to the buyer) then the price of the book will be reasonable, i.e. $low(book) \leq p \leq high(book)$. With this theory we could predict that the situated executed action of *buyer* will be the expected executed action *buy_from(seller1,book1,reasonable_ price)* given the fact that *reasonable_price* $\in$ [*low(book),high (book)*].

If a set of new agents performs actions that satisfy the validated cultural constraints of the group, the problem of their suboptimal behavior with respect to the group is solved. We associate the group of agents whose actions satisfy a validated cultural constraint of another group with no need to know about it, and the group whose actions produced the validated cultural constraint. The relation between groups defined by this association is what we call *implicit culture*. A pair of groups in an implicit culture relation forms an *implicit culture phenomenon*. The actions of a *seller* and a *buyer* are far more effective if they face only offers and demands at reasonable prices, and that is true even if they do not know the cultural constraint.

A system for *implicit culture support* has the goal of establishing an implicit culture phenomenon. It reaches the goal by building validated cultural constraints from observations of situated executed actions, and presenting scenes to the agents such that their expected situated actions satisfy the cultural constraint.

III. An Architecture for Implicit Culture Support

In this section, we present a formal definition of implicit culture, a general architecture for Systems for Implicit Culture Support (SICS) and one example.

A. Basic definitions: scenes, situations and culture

Let *agent_name*, *object_name* and *action_name* be strings. We define: the *set of agents* $\mathcal{P}$ as a set of *agent_name* strings; the *set of objects* $\mathcal{O}$ as a set of *object_name* strings; and the *environment* $\mathcal{E}$ as a subset of the union of the set of agents and the set of objects, i.e., $\mathcal{E} \subseteq \mathcal{P} \cup \mathcal{O}$.

Let E be a subset of the environment ($E \subseteq \mathcal{E}$) and s an *action_name*. We define:

- an *action* α as the pair $\langle s, E\rangle$, where E is the *argument* of α ($E = arg(\alpha)$).

Let $\mathcal{A}$ be a set of actions, $A \subseteq \mathcal{A}$ and $B \subseteq \mathcal{E}$. We define:

- a *scene* σ as a pair $\langle B, A\rangle$ where, for any $\alpha \in A$, $arg(\alpha) \subseteq B$. α is said to be *possible in* σ;
- the *scene space* $\mathcal{S}_{\mathcal{E},\mathcal{A}}$, as the set of all scenes.

Let T be an enumerable and totally ordered set with the minimum t_0. $t \in T$ is said to be a *discrete time*. Let $a \in \mathcal{P}$, α an action and σ a scene:

- a *situation* at the discrete time t is the triple $\langle a, \sigma, t\rangle$. We say that *a faces* the scene σ at time t;
- an *execution* at time t is a triple $\langle a, \alpha, t\rangle$. We say that *a performs* α at time t;
- an action α is a *situated executed action* if there exists a situation $\langle a, \sigma, t\rangle$, where a performs α at the time t and α is possible in σ. We say that a performs α in the scene σ at the time t.

When an agent performs an action in a scene, the environment reacts by proposing a new scene to the agent. The relationship between a situated executed action and a new scene depends on the characteristics of the environment, and in particular on the laws that describe its dynamics. We suppose that it is possible to describe such relationship by an environment-dependent function defined as follows:

$$F_{\mathcal{E}} : A \times \mathcal{S}_{\mathcal{E},\mathcal{A}} \times T \rightarrow \mathcal{S}_{\mathcal{E},\mathcal{A}} \tag{2}$$

Given a situated executed action α_t performed by an agent a in the scene σ_t at the time t, $F_{\mathcal{E}}$ determines the new scene σ_{t+1} ($= F_{\mathcal{E}}(\alpha_t, \sigma_t, t)$) that will be faced at the time $t+1$ by the agent a.

Figure 1 presents how the function $F_{\mathcal{E}}$ works. Particularly, Figure 1.A shows the environment $\mathcal{E}$ in which three agents a, b, and c face the scenes σ_t, σ'_t, and σ''_t respectively (the ellipses indicate the three different situations). At time t the three agents perform respectively the actions α_t, β_t, and γ_t (Figure 1.B). The function $F_{\mathcal{E}}$ changes the scene so that at the time $t+1$ the agents face the scenes σ_{t+1}, σ'_{t+1}, and σ''_{t+1} respectively (Figure 1.C,D).

While $F_{\mathcal{E}}$ is supposed to be a deterministic function, the action that an agent a performs at time t is a random variable $h_{a,t}$ that assumes values in $\mathcal{A}$.

Given an agent $a \in \mathcal{P}$ and a situation $\langle a, \sigma, t\rangle$:

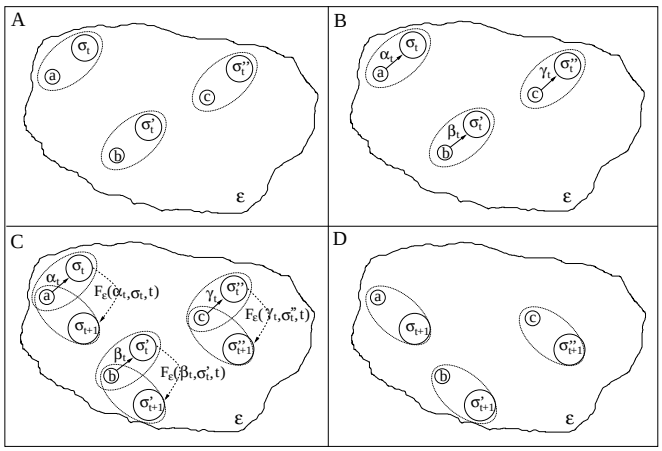

Fig. 1. The environment $\mathcal{E}$

- the *expected action* of the agent a is the expected value of the variable $h_{a,t}$, that is, $E(h_{a,t})$;
- the *expected situated action* of the agent a is the expected value of the variable $h_{a,t}$ conditioned by the situation $\langle a, \sigma, t\rangle$, that is, $E(h_{a,t}|\langle a, \sigma, t\rangle)$.

A set of agents $G \subseteq \mathcal{P}$ is said to be a *group*. Given a group $G = \{a_i\} \subseteq \mathcal{P}$, we denote with the vector $\bar{\alpha}_t = \{\alpha_t[i]\}$ the actions they perform at time t respectively in the scenes $\bar{\sigma}_t = \{\sigma_t[i]\}$. Moreover, we indicate with $\bar{\sigma}_{t+1} = \{\sigma_{t+1}[i]\}$ the vector of the scenes they face after the execution of $\bar{\alpha}_t$ and with $\bar{e}_{t+1} = \{e_{t+1}[i]\}$ the vector of expected situated actions at time $t+1$.

Let $\mathcal{L}$ be a language used to describe the environment (agents and objects), actions, scenes, situations, situated executed actions and expected situated actions. Let $\mathbf{\Sigma}_0$ be an *a priori* theory that describes the environment and the relations among agents and objects in term of actions, scenes, situations and situated executed actions.

Given two groups of agents G and G' we define:

- *cultural constraint theory for* G to be a theory expressed in the language $\mathcal{L}$ that predicates on the expected situated actions of the members of G. If the expected situated actions, estimated by the situated executed actions of G, satisfies the cultural constraint theory, then the theory is said to be *validated*;
- a *cultural action w.r.t.* G to be an executed action that satisfies a validated cultural constraint theory for G;
- *implicit culture* to be a relation $IC(G, G')$ such that G and G' are in the relation if the members of G' execute cultural actions w.r.t. G without knowing the cultural constraint theory for G.
- an *implicit culture phenomenon* to be a pair of groups G' and G related by implicit culture.

Notice that G and G' can be in any relation in terms of inclusion, and just as a special case they can coincide.

B. The architecture

The main goal of a SICS is to establish an implicit culture phenomenon. In the following we propose a general architecture that allows to achieve such a goal by:

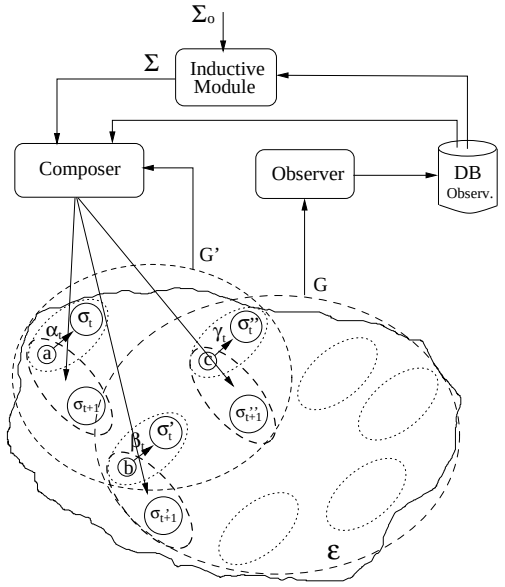

Fig. 2. Architecture

• elaborating a validated cultural constraint theory $\mathbf{\Sigma}$ from a given domain theory and a set of executed situated actions executed by a group G;
• proposing to a group G' a set of scenes such that the expected situated actions of the set of agents G' satisfies $\mathbf{\Sigma}$.

The architecture (Figure 2) consists of the following three basic components:
• an *observer* that stores in a data base (DB Observ.) the situated executed actions of the agents of G.
• an *inductive module* that, using the situated executed actions in DB and the domain theory $\mathbf{\Sigma}_0$, induces a validated cultural constraint theory $\mathbf{\Sigma}$;
• a *composer* that proposes to a group G' a set of scenes $\bar{\sigma}'_{t+1} \neq \bar{\sigma}_{t+1} = \{\sigma_{t+1}[i] = F_{\mathcal{E}}(\alpha_t[i], \sigma_t[i], t)\}$ such that their expected situated actions $\bar{e}_{t+1}$ satisfies $\mathbf{\Sigma}$.

In Figure 2, the composer proposes to the agents a, b, and c the scenes σ_{t+1}, σ'_{t+1}, and σ''_{t+1}, respectively. Notice that in this case the agents b and c belong to both G and G'. This means that also their situated actions are stored in DB and thus they are used to elaborate the theory $\mathbf{\Sigma}$ and the new scenes.

C. Market example

Let us consider an environment *Market* in which there is a set $\mathcal{P}$ of agents (buyers and seller) and a set of objects $\mathcal{O}$ (books and money). Let $\mathcal{A}$ be a set of actions and $\mathcal{S}_{Market,\mathcal{A}}$ be the scene space. In particular, we consider the following actions:
• *ask(o,p)*: asking for object o at price p
• *offer(o,p)*: offering object o at price p
• *buy(x,o,p)*: buying object o from agent x at price p
• *sell(x,o,p)*:selling object o from agent x at price p

Let $\mathbf{\Sigma}_0$ be an a priori domain theory that states that the negotiation between buyer and seller must be successfully concluded in one step. We use for $\mathbf{\Sigma}_0$ the following notation:

$$\begin{array}{l}\forall x, y \in \mathcal{P}, \\ \forall \sigma_x = \langle E_x, A_x\rangle, \sigma_y = \langle E_y, A_y\rangle \in \mathcal{S}_{Market,\mathcal{A}} : \\ \qquad \langle x, \sigma_x, t\rangle \wedge \langle x, ask(o,p1), t\rangle \wedge x \in E_y \wedge \\ \qquad \langle y, \sigma_y, t\rangle \wedge \langle y, offer(o,p2), t\rangle \wedge y \in E_x \rightarrow \\ \qquad \exists \sigma'_x, \sigma'_y \in \mathcal{S}_{Market,\mathcal{A}}, p3 : \\ \qquad \langle x, \sigma'_x, t+1\rangle \wedge \langle x, buy(y,o,p3), t+1\rangle \wedge \\ \qquad \langle y, \sigma'_y, t+1\rangle \wedge \langle y, sell(x,o,p3), t+1\rangle \end{array} \tag{3}$$

that is, for any agents x and y, if at time t, x is facing scene σ_x (which contains y) and it asks for an object o at price $p1$ and y is facing a scene σ_y (which contains x) and it offers the same object for the price $p2 > p1$, then at time $t+1$ there exists a price $p3$ at which x buys object o from y in a scene σ'_x and y sells object o to x in a scene σ'_y.

Let suppose that the inductive module of the SICS realizes that the negotiation between the buyer and seller always takes more than one step, i.e. that $\mathbf{\Sigma}_0$ is no longer satisfied. In order to avoid this, using the situated executed actions of the agents of $G \subseteq \mathcal{P}$, the inductive module induces a cultural constraint theory $\mathbf{\Sigma}$ consistent with $\mathbf{\Sigma}_0$. Let suppose that $\mathbf{\Sigma}$ states that:

$$\begin{array}{l}\forall x, y \in G, \\ \forall \sigma_x = \langle E_x, A_x\rangle, \sigma_y = \langle E_y, A_y\rangle \in \mathcal{S}_{Market,\mathcal{A}} : \\ \qquad \langle x, \sigma_x, t\rangle \wedge \langle x, ask(o,p1), t\rangle \wedge x \in E_y \wedge \\ \qquad \langle y, \sigma_y, t\rangle \wedge \langle y, offer(o,p2), t\rangle \wedge y \in E_x \rightarrow \\ \qquad \exists \sigma'_x, \sigma'_y \in \mathcal{S}_{Market,\mathcal{A}} : \\ \qquad (E(h_{x,t+1}|\langle x, \sigma'_x, t+1\rangle) = buy(y,o,p3)) \wedge \\ \qquad (E(h_{y,t+1}|\langle y, \sigma'_y, t+1\rangle) = sell(x,o,p3)) \wedge \\ \qquad p3 = \frac{9}{10}p2. \end{array} \tag{4}$$

that is, for any agent x and y of G if x asks for an object o at price $p1$ and y offers the same object for the price $p2 > p1$, then the expected situated actions for x and y are of buying object o from y and selling object o to x at $\frac{9}{10}p2$ respectively. Roughly speaking, this means that the buyers and the sellers of G usually agree on a 10% discount. Moreover, $\mathbf{\Sigma}$ says also that the negotiation between buyer and seller takes one step.

Let suppose now that at time $t = 1$ an agent a asks for a *book* for \$100 and an agent $b \in G$ offers the *book* for \$200. In this case, the implicit culture phenomenon is established (and also $\mathbf{\Sigma}_0$ is satisfied) if at the time $t = 2$ agent a buys from b the *book* at \$180 without needing to know that b usually makes a reduction in price of 10%.

In order to do this, at time $t = 1$ the composer observes the two actions performed by a and b and using the situated executed actions in DB composes two scenes σ'_a and σ'_b, respectively for a and b, such that the expected situated actions for a and b satisfy the theory $\mathbf{\Sigma}$. For instance, σ'_a and σ'_b could be two scenes in which a can ask for the *book* for \$180 and b can offer the *book* for \$180, and for which the expected situated actions are:

$$\begin{array}{l} E(h_{a,2}|\langle a, \sigma'_a, 2\rangle) = buy(b, book, 180) \\ E(h_{b,2}|\langle b, \sigma'_b, 2\rangle) = sell(a, book, 180) \end{array}$$

The implicit culture phenomenon is obtained if a buys the *book* from y at \$180, i.e., if a executes cultural actions w.r.t. G. Of course both a and b are always free to decide whether or not to buy or sell the *book*.

In this example, the SICS is used as a mediator between two agents. Even if the mediation does not produce an agreement (i.e., a does not buy the *book* from b at \$180), it has avoided to the two agents to contract the price. The two agents can always start a negotiation, but now starting from \$180.

IV. Instances of Systems for Implicit Culture Support

A SICS based on our architecture enables an agent to perform a more effective behavior in a new environment. For instance, a SICS that intercepts the commands invoked by the users of a system can discover the printers that are used from a set of workstations, and predefine the aliases for a new user. Far from our simple example, instances of SICSs can be found in components of existing systems. In particular, we show that a popular product recommender, a search engine, and a design support system have components that can be considered to be SICSs.

Collaborative Filtering (CF) [4] can be seen as an instance of our architecture. The goal of collaborative filtering is information filtering, namely to extract from a usually long list of items like links or products a small set that matches the preferences of a user. Collaborative filtering reaches the goal exploiting the preferences, expressed actively or passively by other users in terms of ratings. Recommendations are built given the correlations between patterns of ratings on the items.

In this case, the environment $\mathcal{E}$ is composed of items and ratings. The agents belonging to $\mathcal{P}$ are users. An agent can explicitly perform a rating action on an item *express(item1, rating1)* or some other actions like *choose(item1)* or *buy(item2)*, ... etc., that the system assumes to be a rating by associating, for example, *buy(item1)* with *rating1*. We denote the set of these actions by $\mathcal{A}$. The *a priori* domain theory $\mathbf{\Sigma}_0$ in the case of a collaborative filtering system is:

$$\begin{aligned} \forall x \in P, \exists \sigma_x : \forall \sigma'_x \neq \sigma_x \in \mathcal{S}_{\mathcal{E},A} \\ E(h_{x,t}|\langle x, \sigma_x, t\rangle) = express(o, r) \wedge \\ E(h_{x,t}|\langle x, \sigma'_x, t\rangle) = express(o', r') \rightarrow \\ r' < r. \end{aligned} \tag{5}$$

where the scenes σ_x and σ'_x contain o and o' respectively and

$$\begin{aligned} \forall \sigma \in \mathcal{S}_{\mathcal{E},\mathcal{A}},\ \exists K \subseteq G \subseteq \mathcal{P} : \\ \forall x, y \in K\ (x \neq y), \\ E(h_{x,t}|\langle x, \sigma, t\rangle) = E(h_{y,t}|\langle y, \sigma, t\rangle) \end{aligned} \tag{6}$$

The first formula means that given a user there exists a scene such that the rating associated with the expected situated action is a maximum. The second formula expresses the notion that the preferences of the users cluster.

In the case of model-based collaborative filtering, the inductive module characterizes the sets K depending on the situated executed actions and adds the characterizations to $\mathbf{\Sigma}$. Obviously, collaborative filtering algorithms express the characterization in a non-logical form and sometimes not even in a explicit way. Figure 3 shows the architecture in the particular case of memory–based collaborative filtering where no theory is explicitly built. The theory $\mathbf{\Sigma}_0 = \mathbf{\Sigma}$ is directly inserted into the composer.

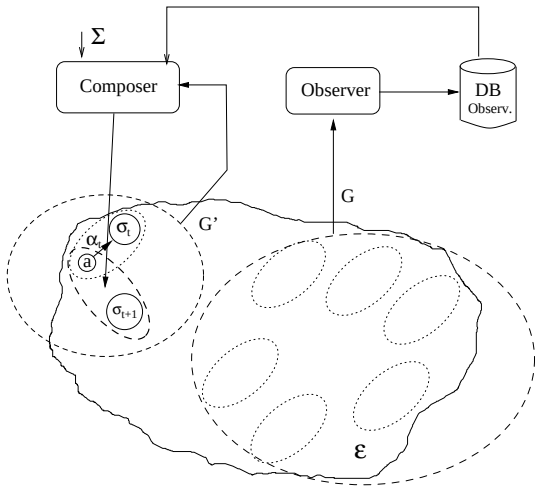

Fig. 3. Memory–Based Collaborative Filtering as a particular case.

Our architecture covers collaborative filtering as a special case. That means that collaborative filtering establishes an implicit culture phenomenon. Leug has already noted that the collaborative filtering changes the social nature of recommendation [5].

A rather popular application of collaborative filtering is exploited in the site `amazon.com`. In this case, the system uses information about book orders of past customers to suggest relevant products when a user is browsing the site. Related to collaborative filtering is the DirectHit (`www.directhit.com/ about/ products/technology_whitepaper.html`) technology for search engines used in popular sites such as `lycos.com` and `hotbot.com`. The search engine intercepts the choices of the bookmarks of the users, given a set of keywords, and uses this information for changing the ranking of bookmarks on future similar searches. The performance of a user is improved by the knowledge of other users in a perfectly transparent way. Finally, the Stamping Advisor system reported by Leake et al. [6] uses a Case Based Reasoning engine in order to provide useful information for supporting stamping design activity in car manufacture. The information is provided proactively with a "just-in-time retrieval" without any need for a request by the user and the cases are collected as a by-product of the user's decision making. The system maps to our architecture because the inductive module is realized by a CBR engine, and the observations of scene and actions does not interfere with the activities of the users.

Our approach generalizes these instances in different directions. First, we pose the implicit culture phenomenon in an agents framework and give the premises for exploiting

it for artificial agents as well. Second, we generalize the forms of cultural constraints. Finally, the general form of SICS supports a group of agents in an integrated way and not only one by one.

V. Related Work

Despite its centrality in *Cultural Anthropology*, the notion of culture resisted several attempts of definition. Following the most accepted definitions, the concept of culture covers almost all the activities that a group of humans undertake a particular geographic area, including material or symbolic production. Obviously, we do not try to address the complete and complex cultural processes of a group of agents but we limit our attention to actions and behaviors. In this regard (i.e., to observe the behavior in order to provide a support) our approach is more related to the use of ethnographic methods for requirements specification [7] rather than to Anthropology *tout court*.

In Artificial Intelligence, there has been some attempts to address cultural issues. Proposed by Reynolds et al. [8] *Cultural Algorithms* concentrate on the aspect of shared knowledge of cultural phenomena. Strongly related to genetic algorithms, *Memetic Algorithms* (see for example [9]) address the problem of evolution of culture in terms of evolution of ideas. The ideas, no matter which is their support, interacts one another and the interaction, via mutation or cross-over phenomena generates different ideas. Finally, there has been the proposal of *Artificial Culture* [10] that can be seen as the natural evolution of the approach of Artificial Life combined with multi-agent systems. The goal is to simulate cultural evolution in an environment in which groups of agents exchange products and communicate with one another.

The main difference with the above approaches is that we are not trying to reproduce a generic cultural phenomena but only a cultural behavior. Moreover, our main issue is not simulative but rather to individuate an effective architecture for improving agent-based systems.

Our architecture is also related to work done in the *Adaptive Interfaces* area and to the notion of situated action that has a long history [11] and has originated strong debates [12]. The user-interface of a system is dynamically changed and the different presentation is guided by the interaction history. We have already shown in Section (IV) how *collaborative filtering* is an instance of the architecture. A strong correlation is also present with the wide area of *User Modeling*. User modeling deals with prototyped users' profiles that are assigned with a user classification process (for an application in e-commerce see for instance [13]). In contrast, our approach does not require classification of agents nor the building of abstract profiles of their interactions. Our contribution is to emphasize the importance of putting into a relation the behaviors of different agents without requiring an explicit effort from them.

VI. Conclusions

We have introduced and defined the notion of implicit culture phenomenon showing how it can be useful in order to guarantee the persistence of a multi-agent system's requirements. We have presented a general architecture for Systems for Implicit Culture Support, namely systems aimed to establish an implicit culture phenomenon on a group of agents. The architecture covers as instances components of existing systems and also suggests further applications with human and artificial agents. The main advantage of SICSs is that they are completely external to the agents and that they can boost their activities and effectiveness without requiring additional computational load.

Acknowledgments

The authors wish to thank Paolo Avesani, Alessandro Ebranati, Paolo Massa, Sabrina Recla, Luciano Serafini and Steven Shapiro for useful discussions.

References

[1] M. Wooldridge, N. R. Jennings, and D. Kinny, "A methodology for agent-oriented analysis and design," in *Proceedings of the Third International Conference on Autonomous Agents*, O. Etzioni, J. P. Muller, and J. Bradshaw, Eds., Seattle, WA, 1998.

[2] Eric Yu, Philippe Du Bois, Eric Dubois, and John Mylopoulos, "From organization models to system requirements – a "cooperating agents" approach," in *Proceedings of the 3rd International Conference on Cooperative Information Systems – CoopIS-95*, Vienna (Austria), 1995.

[3] Eric Yu, *Modelling Strategic Relationship for Process Reengineering*, Technical Report on Research in Data and Knowledge Base Systems, DKBS-TR-94-6, Department of Computer Science, University of Toronto, 1995.

[4] J. Herlocker, J. Konstan, A. Borchers, and J Riedl, "An algorithmic framework for performing collaborative filtering," in *Conference on Research and Development in Information Retrieval*, 1999.

[5] Christopher Lueg, "Considering collaborative filtering as groupware: Experiences and lessons learned," in *Proceedings of the Second International Conference on Practical Aspects of Knowledge Management (PAKM'98)*, Basel, Switzerland, 1998.

[6] D. B Leake, L. Birnbaum, K. Hammond, C. Marlow, and H. Yang, "Integrating information resources: a case study on engineering design support," in *Case–Based Reasoning Research and Development*, Klaus-Dieter Althoff, Ralph Bergmann, and L. Karl Branting, Eds. Springer, Berlin, 1999.

[7] Bonnie A. Nardi, "The use of ethnographic methods in design and evaluation," in *Handbook of Human-Computer Interaction*, M. Helander, T. K. Landauer, and P. Prabhu, Eds., pp. 361–366. Elsevier Science, 1997.

[8] Robert G. Reynolds, "An introduction to cultural algorithms," in *Proceedings of Evolutionary Programming (EP-94)*, San Diego, CA, 1994.

[9] D. Hales, "Selfish memes and selfless agents - altruism in the swapshop," in *Third International Conference on Multi-Agent Systems (ICMAS98)*. 1998, IEEE Computer Society.

[10] Nicholas Gessler, "Artificial culture," in *Fourth International Workshop on the Synthesis and Simulation of Living Systems*, Rodney Brooks and Pattie Maes, Eds., Cambridge, 1994, pp. 430–435, MIT Press.

[11] Lucy A. Suchman, *Plans and Situated Action*, Cambridge University Press, 1987.

[12] N. Kushmerick, "Cognitivism and situated action: two views on intelligent agency," *Computers and Artificial Intelligence*, vol. 15, no. 5, 1996.

[13] L. Ardissono and A. Goy, "Tailoring the interaction with users in electronic shops," in *User Modeling: Proceedings of the Seventh International Conference, UM99*, Judy Kay, Ed., pp. 35–44. Springer, Wien New York, 1999.