Towards a Unified Model of Sociality in Multiagent Systems *

Matthias Nickles Michael Rovatsos Wilfried Brauer Gerhard Weiß

Department of Informatics, Technical University of Munich
D-85748 Garching bei München, Germany
{nickles,rovatsos,brauer,weissg}@cs.tum.edu

Abstract

This paper presents communication systems *(CS) as the first available unified model of socially intelligent systems. It combines the empirical analysis of communication in a social system with logical processing of social information to provide a general framework for computational components that exploit communication processes in multiagent systems. The two main contributions offered by this paper are as follows: First, a formal model of CS that is based on an improved version of expectation networks and their processing is presented. This formal model is based on a novel approach to the semantics of agent communication languages which contrasts with traditional approaches. Second, a number of CS-based applications are described which illustrate the enormous potential and impact of a CS-based perspective of socially intelligent systems.*

Keywords: Artificial Agents, Multiagent Systems, Agent Communication Languages, Agent-Oriented Software Engineering, Socionics

1. Introduction

The crucial property of artificial agents is their *autonomy*, and since *communication* is the only autonomy-preserving way for agents to interact, it can be argued [8, 14] that *any* kind of social relationship among agents (constituted through e.g. virtual organizations, interaction protocols, social norms, common ontologies...) can be described in form of communication structures. Traditional attempts to model the semantics of agent communication languages (ACLs), which constitute communication structures, are mostly based on describing mental states of communicating agents [2, 3, 7, 21] or on public (usually commitment-based) social states [6, 16, 22]. However, both these traditional views fail to recognize that communication semantics evolve during operation of a multiagent system (MAS), and that they always depend on the view of an observer who is tracking the communicative processes in the system [12, 18]. Yet communication dynamics and observer-dependency are crucial aspects of inter-agent communication, especially in the context of open systems in which a pre-determined semantics cannot be assumed, let alone the compliance of agents' behavior with it.

In [8, 1] we have therefore – influenced by sociological systems-theory [9] – introduced *expectations* regarding observable *communications* as a universal means for the modelling of emergent sociality in multiagent systems, and in [18], we have presented – influenced by actor-oriented, socio-cognitive theories [5, 11] – a formal framework for the semantics of communicative action that is *empirical*, *constructivist* and *consequentialist* in nature and analyzed the implications of this model on social reasoning from an agent perspective.

Based upon these works, we suggest that recording observations of message exchange among agents in a multiagent system (MAS) empirically is the only feasible way to capture the meaning of communication, if no *a priori* assumptions about this meaning can be made. Being empirical about meaning naturally implies that the resulting model very much depends on the observer's perspective, and that the semantics would always be the semantics "assigned" to the utterances by that observer, hence this view is inherently constructivist. Since, ultimately, no more can be said about the meaning of a message than that it lies in the expected consequences that this message has, we also adopt a consequentialist outlook on meaning.

In this paper, we present a framework for the formal description of socially intelligent multiagent systems based on the universal, systems-theoretical concept of *communication systems* (CS). Following sociological systems theory, communication systems (also called *social systems*) are systems that consist of interrelated communications which describe their environment [9]. We use this term to denote computational models of such systems that pro-

*This work is supported by Deutsche Forschungsgemeinschaft under contracts no. BR 609/11-2 and MA 759/4-2.

cess empirical information about observed communication which takes place within a MAS of artificial agents (either with the CS as an MAS-external MAS-observer or as a component of an agent which participates in the observed communication himself). Their distinct features are (i) that they only use data about communication for building models of social processes, the underlying assumption being that all relevant aspects of agent interaction are eventually revealed through communication, and (ii) that, if the respective CS is part of an agent, the results of the observations are suitable to take action in the MAS to influence its behavior; in other words, there might be a feedback loop between observation and action, so that an CS-based agent becomes an autonomous component in the overall MAS[1].

CSs might be (part of) socially intelligent software agents. Note, however, that this is not necessarily the case. Although their autonomy presumes some agentification in the traditional sense, their objectives need not be tied to achieving certain goals in the physical (or virtual simulation) world as is the case with "ordinary" agents. Thus, CS are best characterized in a abstract fashion as components used to (trans)form expectations (regardless of how these expectations are employed by agents in their reasoning) and are autonomous with respect to how they perform this generation of expectations.

Thereby, the "semantics" aspect mentioned above plays a crucial role, because the meaning of agent communication lies entirely in the total of communicative expectations in a system [8], and CS capture precisely these expectations and how they evolve.

The remaining sections are structured as follows: We start by introducing expectation networks in section 2, which constitute our formal model for describing communicative expectations. Then, we formally define communication systems and their semantics (section 3). Section 4 discusses possible applications and extensions of the CS, and section 5 concludes.

2. Expectation Networks

Expectations networks (ENs) [8] are the data structures on which communication systems operate. They capture regularities in the flow of communication between agents in a system by interconnecting message templates (nodes) that stand for utterances via links (edges) which are labelled with (i) probabilistic weights called *expectabilities*, (ii) a logical condition and (iii) lists of variable substitutions. Roughly speaking, the semantics of such a weighted edge is as follows: If the variables in the messages have any of the values in the (optional) substitution lists, and the logical condition is currently satisfied, then the weight of this edge reflects the probability with which a message matching the label of the target node is expected to follow the utterance of a message matching the label of the source node of the edge. Before presenting a full definition of ENs, we have to introduce some basic notions and notation we use, and to make certain auxiliary definitions and assumptions. The example network in figure 2 will be used throughout the discussion of ENs to illustrate the purpose of definitions and assumptions.

2.1. Basics

A central assumption that is made in ENs is that observed messages may be categorised as *continuations* of other communications, or may be considered the start of a new interaction that is not related to previous experience. So an edge leading from message m to message m' is thought to reflect the probability of communication being "continued" from the observer's point of view. Usually, continuation depends on temporal and spatial proximity between messages, but it might also be identified through a connection about "subject", or, for example, through the use of the same communication medium (m' was shown on TV after m was shown some time earlier on).

Apart from "ordinary" node labels denoting messages, we use three distinct symbols "$\triangleright$", "$\bot$", and "?". "$\triangleright$" is the label occurring only at the root node of the EN. Whenever a message is considered a start of a new conversation instead of continuing previous sequences, it is appended to this "$\triangleright$"-node. Nodes labelled with "$\bot$" denote that a course of communications is expected to end with the predecessor of this node. The label "?", finally, indicates that there exists no expectation regarding future messages at this node. Nodes with such "don't know" semantics are usually messages that occur for the first time – the observer knows nothing about what will happen after them being uttered.

To define the syntactic details of EN, we introduce formal languages $\mathcal{L}$ and $\mathcal{M}$ used for predicate-logical expressions and for message templates. $\mathcal{L}$ is a simple logical language consisting of propositions $Statement$ potentially containing (implicitly universally quantified) variables and of the usual connectives $\vee$, $\wedge$, $\Rightarrow$ and $\neg$, the logical constants "true" and "false", and braces $()$ for grouping subexpressions together (the language is formally given by the grammar in table 1). Given the set of all possible interpretations $\mathcal{I} = \{I : Statement \to \{\mathrm{true}, \mathrm{false}\}\}$ we define the relation $\models \subseteq \mathcal{I} \times \mathcal{L}$ in the usual way by induction over formulae $\varphi \in \mathcal{L}$ and interpretations $I \in \mathcal{I}$:

$$I \models \varphi \quad \text{iff} \quad \varphi \in Statement \text{ and } I(\varphi) = \mathrm{true}$$
$$I \models \varphi \quad \text{iff} \quad \exists \vartheta : \varphi' \vartheta = \varphi \text{ and } I \models \varphi'$$

[1] For lack of space, we focus on the passive observation of MAS through CSs in this paper. Details on the usage of CSs for the improvement of goal-directed agents can be found in [15].

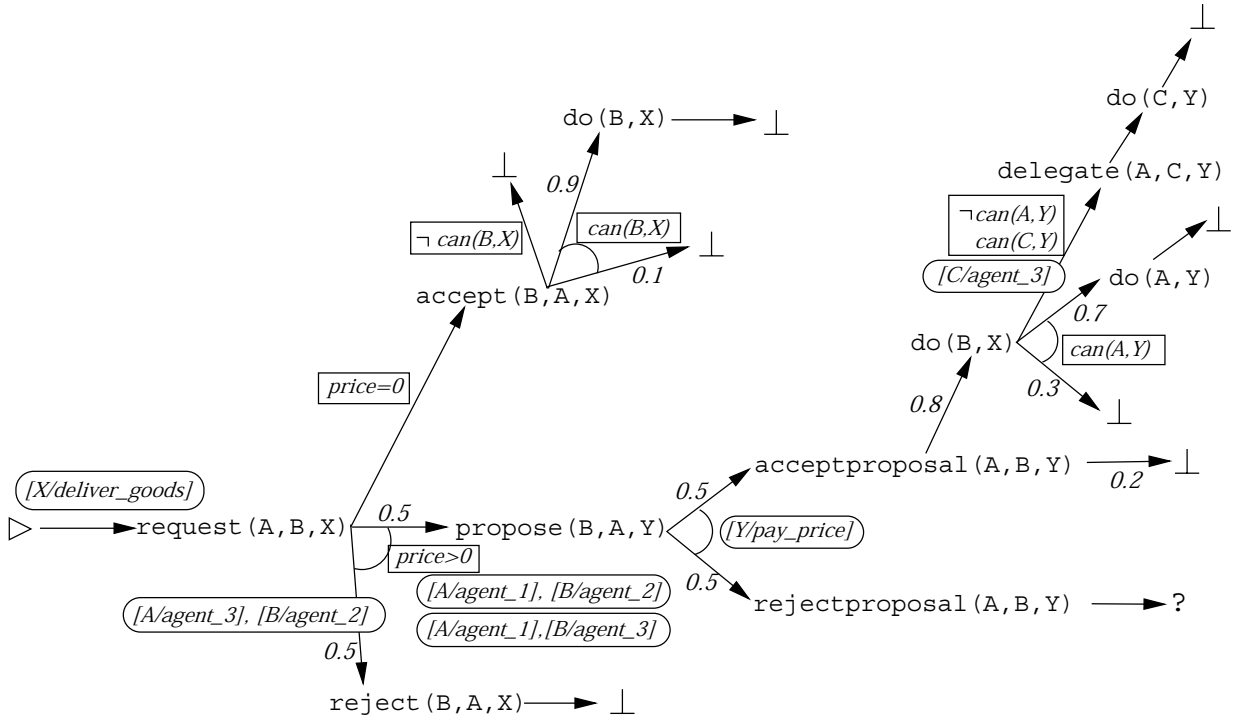

Figure 1.

An expectation network. Nodes are labelled with message templates in `typewriter` *font and the special symbols* $\triangleright$, $\bot$ *and* $?$*; they are connected by (solid) edges labelled with numerical expectabilities in italic font. Substitution lists/conditions belonging to edges appear in rounded/edged boxes near the edge. If neighbouring edges share a condition this is indicated by a drawn angle between these edges. This example network shows a short communication course of three agent roles,* A*,* B *and* C*, which are instantiated with three agents through substitution lists (*$A = agent\_1$*,* $B = agent\_2$*,* $C = agent\_3$*). The provision of such substitution lists is optional. The EN starts with a request to do action* X *(*$deliver\_goods$*) of* A *directed to* B*. In case condition* $price = 0$ *is fulfilled,* B *is expected to always accept this request and to perform the requested action* X *in case he is able to do it (condition* $can(B, X)$ *is true). Otherwise, with probability 0.5 the request will be rejected and the communication ends (*$\bot$*). With probability 0.5,* B *answers with a proposal* Y *(where* $Y = pay\_price$*), which is accepted by* A *with probability 0.5. After rejection, the further course of communication is unknown (*$?$*), whereas the acceptance leads to the fulfillment of* X *through* B *(*$do(B, X)$*). In the latter case,* A *in turn does* Y *(i.e., pays the price for* X*), or delegates this to* C *if he is not able to pay (*$can(A, Y)$ *is false).*

$$I \models \neg\varphi \quad \text{iff} \quad I \not\models \varphi$$
$$I \models \varphi \vee q \quad \text{iff} \quad I \models \varphi \text{ or } I \models q$$

where $\vartheta = \langle [v_1/t_1], \ldots, [v_k/t_k] \rangle$ is a variable substitution. As usually, $\wedge$ and $\Rightarrow$ can be defined as abbreviations through the other operators. Also, we write $\models \varphi$ if φ is a tautology that is satisfied by any $I \in \mathcal{I}$. A *knowledge base* $KB \in 2^{\mathcal{L}}$ can be any finite set of formulae from $\mathcal{L}$. For simplicity, we will often write $KB \models \varphi$ to express $\models (\bigwedge_{\varphi' \in KB} \varphi' \Rightarrow \varphi)$.

As for $\mathcal{M}$, this is a formal language that defines the message patterns used for labelling nodes in expectation networks. Its syntax is given by the grammar in table 1. Messages observed in the system (we write $\mathcal{M}_c$ for the language of these *concrete* messages) can be either physical messages of the format $\texttt{do}(a, ac)$ where a is the executing agent and ac is a symbol used for a physical action, or a non-physical message $performative(a, b, c)$ sent from a to b with content c. (Note that the symbols used in the $Agent$ and $PhysicalAction$ rules might be domain-dependent symbols the existence of which we take for granted.) The node labels (type $MsgPattern$) used in the expectation networks may also contain variables for agents and physical actions (though not for performatives). Following the concept of agent roles introduced in [8, 1], the variables for agents are called (agent) *roles*. These variables are useful to generalize over different observed messages, and can optionally be further specified by adding variable substitution lists. The content c of a non-physical

$$
\begin{array}{rcl}
Var & \rightarrow & X \mid Y \mid Z \mid \ldots \\
AgentVar & \rightarrow & A_1 \mid A_2 \mid \ldots \\
PhysicalActVar & \rightarrow & X_1 \mid X_2 \mid \ldots \\
Expect & \in & [0;1] \\
Agent & \rightarrow & agent\_1 \mid \ldots \mid agent\_n \\
Head & \rightarrow & it\_rains \mid loves \mid \ldots \\
Performative & \rightarrow & accept \mid propose \mid reject \mid inform \\
 & & \mid \ldots \\
PhysicalAction & \rightarrow & move\_object \mid pay\_price \\
 & & \mid deliver\_goods \mid \ldots \\
Message & \rightarrow & Performative(Agent, Agent, LogicalExpr) \\
 & & \mid do(Agent, Agent, PhysicalAction) \\
MsgPattern & \rightarrow & Performative(AgentTerm, AgentTerm, \\
 & & LogicalExpr) \\
 & & \mid do(AgentTerm, AgentTerm, \\
 & & PhysicalActTerm) \\
 & & \mid \triangleright \mid \bot \mid ? \\
PhysicalActTerm & \rightarrow & PhysicalActVar \mid PhysicalAction \\
AgentTerm & \rightarrow & AgentVar \mid Agent \\
LogicalExpr & \rightarrow & (LogicalExpr \Rightarrow LogicalExpr) \\
 & & \mid (LogicalExpr \vee LogicalExpr) \\
 & & \mid (LogicalExpr \wedge LogicalExpr) \\
 & & \mid \neg LogicalExpr \\
 & & \mid Statement \\
Statement & \rightarrow & Head \mid Head(TermList) \mid \text{true} \\
 & & \mid \text{false} \\
TermList & \rightarrow & TermList, Term \mid Term \\
Term & \rightarrow & Var \mid AgentTerm \mid MsgPattern \\
 & & \mid Graph \\
EdgeList & \rightarrow & (MsgPattern, Expect, MsgPattern, \\
 & & LogicalExpr, SubstList)\ EdgeList \mid \varepsilon \\
Graph & \rightarrow & \langle EdgeList \rangle \\
SubstList' & \rightarrow & SubstList'\ Subst \mid \varepsilon \\
SubstList & \rightarrow & \langle SubstList' \rangle \\
Subst & \rightarrow & [AgentVar/Agent] \\
 & & \mid [PhysicalActVar/PhysicalAction] \\
 & & \mid [Var/Term]
\end{array}
$$

Table 1.

A grammar for messages, generating the languages $\mathcal{M}$ (the language of message patterns, using $MsgPattern$ as starting symbol), $\mathcal{M}_c$ (the language of concrete messages, using $Message$ as starting symbol) and the logical language $\mathcal{L}$ (using $LogicalExpr$ as starting symbol).

action, finally, is given by type $LogicalExpr$. It can either be (i) an atomic proposition, a (ii) message term or physical action term, (iii) an expectation network[2], or (iv) a logical formula containing these elements according to table 1. Syntactically, expectation networks are here represented as lists of edges (m, p, n, c, l) where m and n are message terms, p is a transition probability (expectability) from m to n, c is a logical condition, l is a list of variable substitutions. We use functions $in : V \rightarrow 2^C$, $out : V \rightarrow 2^C$, $source : C \rightarrow V$ and $target : C \rightarrow V$ which return the ingoing and outgoing edges of a node and the source and target node of an edge, respectively, in the usual sense. C is the set of all edges, V the set of all nodes in the EN. $cond : C \rightarrow \mathcal{L}$ returns the conditions of edges, $subst : C \rightarrow SubstList$ (with $SubstList$ as in table 1) returns the edges' substitution lists. Edges denote correlations in observed communication sequences. Each cognitive edge is associated with an expectability (returned by $Expect : C \rightarrow [0;1]$) which reflects the probability of $target(e)$ occurring shortly after $source(e)$ in the same communicative context (i.e. in spatial proximity, between the same agents, etc.).

The full meaning of these ingredients will be further clarified once the full definition of expectation networks has been presented. Note that according to table 1 expectation networks are allowed to be contained within message terms of expectation network nodes themselves to allow the modelling of the communication of complex expectation structures among agents.

2.2. Edge Conditions

As a final ingredient to network edges, we briefly discuss edge conditions. The idea is that these conditions should further define the scope of validity to cases in which a formula can be shown to hold using the observer's knowledge base. So, if $\varphi = cond(e)$, then e is only relevant iff $KB \models \varphi$.

Because all conditions for outgoing edges of a certain node should be mutually exclusive to ensure that later the semantics of a certain message trajectory can be calculated unambiguously, we want the sum of expectabilities of all out-edges of a node to be one for a certain knowledge base content[3]. In other words, the condition

$$\forall v \sum_{e \in out(v), KB \models cond(e)} Expect(e) = 1$$

should hold.

This can be ensured, for example, by guaranteeing that the following condition holds through appropriate construction rules for the EN. Assume the outgoing links $out(V)$ of every node v are partitioned into sets $O_1, O_2, \ldots O_k$ where the links' expectabilities in each O_i are non-negative and sum up to one[4]. Now let all

[2] Such expectation networks within messages are useful to replace performatives of agent communication languages. We have to refer to [12] for details of this concept.

[3] From a probabilistic point of view, it would be sufficient to demand a sum lower or equal one, but a sum of exactly one (which is practically always feasibly through insertion of a "dummy" '?'-edge) formally ensures the exhaustiveness of the set of outgoing edges.

[4] Formally, $out(v) = \cup_{1 \leq i \leq k} O_i$ and $\forall 1 \leq i < j \leq k. O_i \cap O_j = \emptyset$, and $\forall i \leq k. (\forall o \in O_i. Expect(o) \geq 0 \wedge \sum_{o \in O_i} Expect(o) = 1)$.

edges in O_i share the same edge condition, i.e. $\forall i \exists \varphi \forall o \in O_i.(cond(o) = \varphi)$ and define $cond(O_i)$ as precisely this shared condition φ. (The O_i sets are precisely those subsets of $out(v)$ connected by a drawn angle in figure 2.)

If we make sure that the outgoing links of every node are partitioned in this way, we can assign mutually exclusive conditions to them, i.e. ensure that

$$\forall i \neq j.cond(O_i) \wedge cond(O_j) \equiv \text{false}$$
$$\text{and} \quad \vee_i \, cond(O_i) \equiv \text{true}$$

This way, it is not only guaranteed that we can derive unambiguous probabilities directly from the *Expect* values, but also that we can do so for *any* knowledge base contents (cf. 2.4)[5].

2.3. Formal Definition

Having discussed all the prerequisites, we can now define ENs formally:

Definition 1. An *expectation network* is a structure

$$EN = (V, C, \mathcal{M}, \mathcal{L}, H, mesg, cond, subst, Expect)$$

where

- V with $|V| > 1$ is the set of nodes,
- $C \subseteq V \times V$ are the *cognitive* edges (or edges for short) of EN. (V, C) is a tree called *expectation tree*.
- $\mathcal{M}$ is a *message term language*[6], $\mathcal{L}$ is a logical language, $cond : C \to \mathcal{L}$ returns the conditions of edges,
- $mesg : V \to \mathcal{M}$ is the *message label* function for nodes such that
  - $mesg(v) = \triangleright$ exactly for the root node of (V, C),
  - $\forall v \in V. \forall e, f \in out(v).$ $\neg unify(mesg(target(e)), mesg(target(f)))$ (where $unify$ shall be $true$ iff its arguments are syntactically unifiable, i.e., target node labels of outgoing links never match),
- $H \in \mathbb{N}$ is a finite *communication horizon*,
- $Expect : C \to [0; 1]$ returns the edges' expectabilities,
- $subst : C \to SubstList$ (with $SubstList$ as in table 1) returns the edges' substitution list.

[5] This comes at the price of having to insert redundant edges in some situations. For example, insertion of a new edge e with $cond(e) = \varphi$ if $out(v) = \emptyset$ necessitates insertion of another edge e' with $cond(e) = \neg\varphi$.

[6] All languages as defined in the previous sections.

Our full formal framework [15] also defines so-called *normative* edges, which have been omitted here for lack of space. In contrast to cognitve edges, the expectabilities of normative edges are only seldomly adapted by the communication system according to newly observed messages, and under very specific circumstances. In the tradition of systems theory, here the term "cognitive" means "adaptable through cognition about observations".

The only element of this definition that has not been discussed so far is the communication horizon H, which denotes the scope of maximal message sequence length for which the EN is relevant. It is necessary for defining the semantics of the EN, and will be discussed in detail in the following section.

2.4. Formal Semantics of Message Sequences

The purpose of an EN is to provide a semantics for messages. For an arbitrary set S, let $\Delta(S)$ be the set of all (discrete) probability distributions over S with finite support. We define the semantics $I_{EN}(KB, w)$ of an observed message sequence w in a network EN as a mapping from knowledge base states and current message sequence prefixes to the posterior probability distributions over all possible postfixes (conclusions) of the communication. Formally,

$$I_{EN}(KB, w) = f_w, \quad f_w \in \Delta(\mathcal{M}_c^*) \tag{1}$$

where

$$f_w(w') = \frac{g_w(w'\bot)}{\sum_{v \in \mathcal{M}_c^*} g_w(v\bot)} \tag{2}$$

is defined as the normalized value of $g_w(w'\bot)$. $g_w(w'\bot)$ represents the probability that w will be concluded by message sequence w', for any $w, w' \in \mathcal{M}^*$. We compute the probability for $w'\bot$ to make sure w' is followed by a node with label $\bot$ in the network, because the probability of w' is the probability with which the communication sequence will *end* after $w'_{|w'|}$ (and not that w' will simply be the prefix of some longer sequence). Also note that the sum in the denominator is not, as it may seem, infinite, because f_w has finite support and the length of the considered message sequences is limited by means of the communication horizon H (see below), and that the semantics of w depends on KB, because only those edges which have conditions that are true according to KB are used for calculating the semantics of w.

Informally, the probability of w' should be inferred from multiplying all the expectability weights along the path that matches w' (if any). Before presenting the top-level formula for $g_w(w')$, we need some auxiliary definitions:

Firstly, we need to determine the node in a network EN that corresponds to a word w, which we denote by

$mesg^{-1}$:

$$mesg^{-1}(\varepsilon) = v \quad :\Leftrightarrow \quad mesg(v) = \triangleright$$

$$mesg^{-1}(wm) = \begin{cases} v' & \text{if } \exists (v,v') \in C(KB). \\ & \exists \vartheta \in subst((v,v')). \\ & (mesg(v') \cdot subst(w)\vartheta = m \\ & \wedge mesg^{-1}(w) = v) \\ \bot & \text{if no such } v' \text{ exists} \end{cases} \tag{3}$$

if $w \in \mathcal{M}_c^*$, $m \in \mathcal{M}_c$[7]. The first case states that the node corresponding to the empty sequence ε is the unique root node of (V, C) labelled with $\triangleright$. According to the second case, we obtain the node v' that corresponds to a sequence wm if we take v' to be the successor of v (the node reached after w) whose label matches m under the following condition:

There has to be a substitution $\vartheta \in subst((v,v'))$ which, when composed with the substitution $subst(w)$ applied so far to obtain the messages in w_1 to $w_{|w|}$ from the respective nodes in EN, will yield m if applied to $mesg(v')$. This is expressed by $mesg(v') \cdot subst(w)\vartheta = m$. In other words, there is at least one combined (and non-contradictory) variable substitution that will make the node labels along the path $mesg^{-1}(wm)$ yield wm if it is applied to them (concatenating substitutions is performed in a standard fashion). Thereby, the following inductive definition can be used to derive the substitution $subst(w)$ for an entire word w:

$$w = \varepsilon : \quad subst(w) = \langle\rangle$$
$$w = w'm : \quad subst(w) =$$
$$subst(w') \cdot unifier(mesg(mesg^{-1}(wm)), m)$$

where $\cdot$ is a concatenation operator for lists and $unifier(\cdot,\cdot)$ returns the most general unifier for two terms (in a standard fashion). Thus, $subst(w)$ can be obtained by recursively appending the unifying substitution of the message label of each node encountered on the path w to the overall substitution. With all this, we are able to compute $g_w(w')$ as follows:

$$g_w(w') = \begin{cases} |\cup_{i=1}^{H} \mathcal{M}_c^i|^{-1} \\ \text{if } \exists v \in out(mesg^{-1}(w)).mesg(v) =? \\ \prod_i \left(\sum_{e \in pred(ww',i)} S(e)\right) & \text{else} \end{cases} \tag{4}$$

which distinguishes between two cases: if the path to node $mesg^{-1}(w)$ whose labels match w (and which is unique, because the labels of sibling nodes in the EN never unify) ends in a "?" label, the probability of a w' is simply one over the size of all words with length up to the communication horizon H (hence its name). This is because the semantics of "?" nodes is "don't know", so that all possible conclusions to w are uniformly distributed. Note that this case actually only occurs when new paths are generated and it is not known where they will lead, and also that if an outgoing link of a node points to a node with label "?", then this node will have no other outgoing links.

In the second case, i.e. if there is no "?" label on the path p from $mesg^{-1}(w)$ to $mesg^{-1}(ww')$, then the probability of w' is the product of weights $S(e)$ of all edges e on p. Thereby, $S(e)$ is just a generalized notation for expectability or normative force depending on the typed edge, i.e. $S(e) = Exp(e)$ for $e \in C$. The sum of these S-values is computed for all ingoing edges $pred(ww', i)$ of the node that represents the the ith element of w', formally defined as

$$\forall w \in \mathcal{M}_c^*.pred(w,i) = \begin{cases} in(mesg^{-1}(w_1 \cdots w_i)) \\ \text{if } mesg^{-1}(w_1 \cdots w_i) \neq \bot \\ \emptyset & \text{else} \end{cases} \tag{5}$$

3. Communication Systems

A communication system can be seen as a description of the social dynamics of a multiagent system. The two main purposes of a CS are i) to capture the social expectations (represented as an EN) in the current state of a multiagent system under observation, and ii) to capture changes to these expectation structures. Whereas the EN models the current meaning of communicative action sequences (i.e., their expected, generalized continuations in a certain context of previous message utterances), the CS models the way the EN is build up, and, if necessary, adapted according to new statistical observations. As already mentioned in section 1, in contrast to agents who reason about expectations (such as *InFFrA* agents [20]), a CS need not necessarily be an active agent who takes action in the MAS itself.

Describing how communication systems work should involve (at least) clarifying:

- which communicative actions to select for inclusion in an EN,
- where to add them and with which expectability (in particular, when to consider them as "non-continuations" that directly follow "$\triangleright$"),
- when to delete existing nodes and edges (e.g. to "forget" obsolete structures), and how to ensure integrity constraints regarding the remaining EN.

[7] For convenience, let $C(KB)$ be the set of nodes within the subnetwork of expectation network EN where the edge set is reduced to those edges whose conditions are satisfied under KB.

A formal framework for specifying the details of the above is given by the following, very general, definition:

Definition 2. A *communication system* at time t is a structure

$$CS_t = (\mathcal{L}, \mathcal{M}, f, \varpi_t, \kappa)$$

where

- $\mathcal{L}$, $\mathcal{M}$ are the formal languages used for logical expressions and messages (according to table 1),
- $f : \mathcal{EN}(\mathcal{L}, \mathcal{M}) \times \mathcal{M}_c \rightarrow \mathcal{EN}(\mathcal{L}, \mathcal{M})$ is the *expectation structures update function* that transforms any expectation network EN to a new network upon experience of a message $m \in \mathcal{M}_c$,
- $\varpi_t = m_0 m_1 ... m_t \in \mathcal{M}_c^*$ is the list of all messages observed until time t. The subindexes of the m_i impose a linear order on the messages corresponding to the times they have been observed[8].
- $\kappa : 2^{\mathcal{L}} \times \mathcal{M}_c \rightarrow 2^{\mathcal{L}}$ is a *knowledge base update function* that transforms knowledge base contents after a message accordingly,

and $\mathcal{EN}(\mathcal{L}, \mathcal{M})$ is the set of all possible expectation networks over $\mathcal{L}$ and $\mathcal{M}$. The intuition is that a communication system can be characterized by how it would update a given knowledge base and an existing expectation network upon newly observed messages $m \in \mathcal{M}_c$. The EN within CS_t can thus be computed through the sequential application of the expectation structures update function f for each message within ϖ, starting with an empty expectation network (i.e., an EN which contains only the start node). ϖ_{t-1} is called the *context* of the message observed at time t, and $I_{EN}(KB, \varpi_t)$ computes the semantics of this message within this context.

This definition of CS is very general, as it does not prescribe how the EN is modified by the CS. However, some assumptions are reasonable to make (although not obligatory):

- If KB is the current knowledge base, $\kappa(KB, m) \models KB(m)$ should hold, so that all facts resulting from execution of m are consistent with the result of the κ-function.
- An EN should predict the future of the respective observable communication sequences as accurately as possible. Although there is no canonical method a CS should use to construct and update ENs, we propose the following very general heuristic: if any message sequence w' has occurred with frequency $\Pr(ww')$ as a continuation of w in the past, and EN' is the same as EN, $I_{EN'}(KB, w)(w') = \Pr(ww')$ should be the case, i.e. the expectabilities along a certain path within the expectation tree shall reflect the frequencies with which the respective message sequences have been occurred.

In addition to these basic assumptions, we propose the following functionality a CS shall provide to be of practical use:

3.0.1. Message Filtering and Syntax Recognition. Depending on its goals and the application domain, the CS as an autonomous observer might not be interested in all observable messages. Since ENs may not provide for *a priori* expectations, the discarding of such "uninteresting" messages can only take place *after* the semantics (i.e., the expected outcome) of the respective messages has already been derived from previous observation. Because discarding messages bears the risk that these messages become interesting afterwards, as a rule of thumb, message filtering should be reduced to a minimum. More particularly, messages should only be filtered out in cases of more or less settled expectations. Paying attention to every message and filtering uninteresting or obsolete information later by means of structure reweighting and filtering (cf. below) is presumably the more robust approach.

3.0.2. Structure Expansion and Generalization. Structure expansion is concerned with the growth of an EN in case a message sequence is observed which has no semantics defined by this EN yet. In order to do so, we could start with an empty EN and incrementally add a node for each newly observed message. But this would be not smart enough, because it does not take advantage of generalizable message sequences, i.e. different sequences that have approximately the same meaning. In general, such a generalization requires a relation which comprises "similar" sequences. The properties of this relation of course depends on domain- and observer-specific factors. A quite simple way of generalizing is to group messages which can be unified syntactically, using the message patterns introduced in table 1.

In theory, the expansion of the EN would never be necessary if we could a-priori generate a *complete EN*, i.e. an EN which contains dedicated paths for *all* possible message sequences. In this case, the CS would just have to keep track of the perceived messages using ϖ and to identify this sequence within the EN to derive its semantics, with no need for f. For obvious reasons, such a complete EN cannot be constructed in practice.

3.0.3. Pruning the EN. Several further methods of EN processing can be conceived of that aid in keeping the computation of (approximate) EN semantics tractable. This

[8] For simplicity, we assume a discrete time scale with $t \in \mathbb{N}$, and that no pair of messages can be uttered at the same time.

can be achieved by continuously modifying expectation structures using certain meta-rules, for example:

1. "fading out" old observations by levelling their edge weights;
2. replacing large sets of sibling edges with (approximately) uniformly distributed expectabilities with single edges leading to "?" nodes;
3. removal of "?"s that are not leafs. Such nodes can occur as outcome of the previous measure.
4. keeping the EN depth constant through removal of one old node for each new node to save space and remove obsolete structures (e.g. using the communication horizon H as maximum EN depth);
5. removal of edges with very low expectabilities. In case this results in cut-off branches, these have to be connected with the start node subsequently.

Since these modifications are highly application-dependent, we don't provide exact criteria for their practical application here.

4. Applications and Extensions

The modelling of social structures on the basis of expectation networks and communication systems allows for novel approaches to a variety of challenging issues in multiagent system technology. In the following, we review three of these issues, namely, (i) identification of ontologies for inter-agent communication and – closely related – the finding of verifiable and flexible semantics for agent communication languages; (ii) *mirror holons* as a new model for holonic theories of agency and software engineering methods based on expectation-oriented modelling and analysis of multiagent systems; (iii) the agent-level social reasoning architecture *InFFra*.

4.1. Social Ontologies

In Distributed Artificial Intelligence (DAI), an ontology is a set of definitions as a means to provide a common ground in the conceptual description of a domain for communication purposes. Ontologies are usually represented as graphical hierarchies or networks of concepts, topics or classes, and either top-down imposed on the agents or set up bottom-up by means of ontology negotiation. In a similar way, expectation networks are descriptions of the social world in which the agents exist. But ENs do not only describe social (i.e. communication) structures, but indirectly also the communication-external environment the message content informs about. Thus, communication systems can be used, in principle, for an incremental collection of ontological descriptions from different autonomous sources, resulting in stochastically weighted, possibly conflicting, competitive and revisable propositions about environmental objects. The crucial difference to traditional mechanisms is that such a *social ontology* represents expectations about how a certain object will be described in future communication. This opposes the "imposed ontologies" view somewhat, where the ontology provides an *a priori* grounding for communication, and makes this approach appear particularly suitable for open multiagent systems with a highly dynamic environment, where homogenous perception among agents cannot be assumed. Also, it is appropriate whenever descriptions are influenced by individual preferences such that a consensus cannot be achieved (think, e.g., about "politically" biased resource descriptions in the context of the Semantic Web [13]). In the following, we'll sketch two approaches for extracting social ontologies from expectation networks.

4.1.1. Extraction of Speech Act Types. The current version of FIPA-ACL [4] provides an extensible set of speech-act performative types with semantics defined in a mentalistic fashion. In our approach, we can imagine a special CS variant as a MAS component (e.g., a so-called multiagent system *mirror* [14, 8]) that provides the agents with a set of performatives *without* any predefined semantics and wait for the semantics of such "blank" performatives to emerge. To become predictable, it is rational for an agent to stick to the meaning (i.e., the consequences) of performatives, at least to a certain extent. This meaning has been previously (more or less arbitrarily) "suggested" for a certain performative by some agent performing demonstrative actions after uttering it.

Of course, a single agent is usually not able to define a precise and stable public meaning for these performatives, but at least the intentional attitude associated with the respective performative needs to become common ground for communication to facilitate a non-nonsensical, non-entropic discourse [18, 12]. A particular performative usually appears at multiple nodes within the EN, with different consequences at each position, depending on context (especially on the preceding path), message content and involved sender and receiver. To build up an ontology consisting of performative types, we have to continually identify and combine the occurrences of a certain performative within the current EN to obtain a general meaning for this performative (i.e., a "type" meaning). Afterwards, we can communicate this meaning to all agents using some technical facility within the multiagent system, like a MAS mirror or an "ACL semantics server". Of course, such a facility cannot impose meaning in a normative way as the agents are still free to use or ignore public meaning as they like, but it can help to spread language data like a dictio-

nary or a grammar does for natural languages. The criteria for the identification and extraction of performative meaning from ENs are basically the same as the criteria we proposed in 3 for the generalization over message sequences.

4.1.2. Extraction of Domain Descriptions. While a set of emergent speech act types constitutes a social ontology for communication events, classical ontologies provide a description of an application domain. To obtain a social version of this sort of ontology from an EN, two different approaches appear to be reasonable: (1) Inclusion of environment events within the EN and (2) probabilistic weighting of assertions. The former approach, which is introduced in [12], treats "physical" events basically as utterances. Similar to the communicative reflection of agent actions by means of `do`, a special performative `happen`$(event)$ would allow EN nodes that reflect events occurring in the environment. These events will be put in the EN either by a special CS which is able to perceive the agents' common environment, or by the agents themselves as a communicative reflection of their own perceptions. A subset of $event$ is assumed to denote events with consensual semantics (think of physical laws), i.e., the agents are not free to perform an arbitrary course of action after such an event has occurred, whereas the remainder of $event$ consists of event tags with open semantics that has to be derived empirically from communications observation just as for "normal" utterances. If such an event appears for the first time, the CS does not know its meaning in terms of its consequences. Its meaning has thus to be derived aposteriori from the communicational reflection of how the agents react to its occurrence. In contrast, approach (2), which we proposed for the agent-based competitive rating of web resources [13], exploits the propositional attitude of utterances. The idea is to interpret certain terms within $LogicalExpr$ as domain descriptions and to weight these descriptions according to the amount of consent/dissent (using predefined performatives like $Assert$ and $Deny$). The weighted propositions are collected within a knowledge base (e.g., KB as defined before) and are communicated to the agents in the same way as the emergent speech act types before. Unlike approach (1), ontologies are constructed "by description" not "by doing" in this way. The advantage of approach (1) lies in its seamless integration of "physical" events into the EN, whereas (2) is probably more easy to apply in practice.

4.2. Mirror Holons: Multi-Stage Observation, Reflection and Enactment of Communication Structures

In [8, 1], we have introduced the *social system mirror* architecture for open MAS. The main component of this architecture is a so-called social system mirror (or "mirror" for short), a middle agent containing a CS which continually observes communications, empirically derives emergent expectation structures (represented as an ENs, which might also contain *normative structures*, which are given by the designer instead of being learned from statistical observations) from these observations, and "reflects" these structures back to the agents. In addition to the recording of empirical expectation structures described in this work, the mirror is a goal-directed agent within the MAS. Its goals are to influence agent behavior by means of system-wide propagation of social structures and norms to achieve quicker structure evolution (catalysis) and higher coherence of social structures without restricting agent autonomy, and the provision of a representation of a dynamic communication system for the MAS designer. While a mirror only models a single communication system, and, except for the propagation of expectations, does not take action itself, the successor architecture *HoloMAS* [14] is able to model multiple communication systems at the same time through multiple *mirror holons* in order to model large, heterogenous systems. In addition, a mirror holon can take action himself by means of the execution of social programs which are generated from emergent expectation structures. "Ordinary agents" (and other mirror holons) can optionally be involved in this execution process as *effectors*, which realize holon commands within their physical or virtual application domain (unless they deny the respective command). In any case they can influence the social programs within a mirror holon through the irritation of expectation structures by means of communication. A mirror holon thus represents and (at least to some extent) replaces the functionality of the ordinary agents that contribute to the emergence of the respective expectation structures, but it does not disregard the autonomy of his adjoint actors. Another difference between mirror holons and traditional agent holons is that a mirror holon does not represent or contain groups of agents, but instead a certain functionality which is identified in form of regularities in the observed communications. This functionality is extracted and continually adopted from dynamic expectation structures regarding criteria like consistency, coherence and stability, corresponding to the criteria sociological systems theory ascribes to social programs [9]. Mirror holons pave the way for applications in which agent autonomy should not (or cannot) be restricted on the one hand, while reliable, time-critical system behavior is desired. They can also be used as representatives for entire communication systems (e.g., virtual organizations) that behave smoothly towards third parties whenever the communication system itself lacks coherence due to, for example, inner conflicts.

4.2.1. Expectation-Oriented Software Development. As it has been recognized that due to new requirements

arising from the complex and distributed nature of modern software systems the modularity and flexibility provided by object orientation is often inadequate and that there is a need for encapsulation of robust functionality at the level of software components, agent-oriented approaches are expected to offer interesting prospectives in this respect, because they introduce interaction and autonomy as the primary abstractions the developer deals with.

However, although interaction among autonomous agents offers great flexibility, it also brings with it contingencies in behavior. In the most general case, neither peer agents nor the MAS designer can "read the mind" of an autonomous agent, let alone change it. While the usual strategy to cope with this problem is to restrict oneself to closed systems, this means loosing the power of autonomous decentralized control in favour of a top-down imposition of social regulation to ensure predictable behavior. The EXPAND method (Expectation-oriented Analysis and Design) [1] follows a different approach. EXPAND is based on expectations networks as a primary modelling abstraction which both system designer and agents use to manage the social level of their activities. This novel abstraction level is made available to them through a special version of the social system mirror (4.2), i.e., a special CS, very similar to a CASE tool. For the designer, this mirror acts as an interface he uses to propagate his desired expectations regarding agent interaction to the agents and as a means for monitoring runtime agent activity and deviance from expected behavior. For agents, this mirror represents a valuable "system resource" they can use to reduce contingency about each other's behavior. EXPAND also describes an evolutionary process for MAS development which consists of multiples cycles: the modelling of the system level, the derivation of appropriate expectation structures, the monitoring of expectation structure evolution and the refinement of expectation structures given the observations made in the system. For a lack of space, we have to refer the interested reader to [1] for details.

4.3. Social reasoning with InFFrA

The Interaction Frame and Framing Architecture InFFrA [20] is a social reasoning architecture in which so-called *interaction frames* are used to represent patterns of social interaction and strategically employed by socially intelligent agents to guide their interaction and communication behaviour. This is achieved by agents deriving models of frames from observation of encounters and applying the most appropriate patterns in future interactions (this process is called *framing*). The concepts of frame and framing are based on Erving Goffman's micro-social analyses of everyday life [5].

From a CS perspective, InFFrA is nothing but an *agent-centric* interpretation of our concepts. Instead of defining a general observer of communication, InFFrA exclusively deals with agent observers, and rather than observing general communication processes, InFFrA agents only observe "face-to-face" interaction processes (mostly those they are personally involved in).

In other words, interaction frames in InFFrA are micro-models of communicative expectations that encode knowledge about communication processes from the standpoint of an agent observer. What makes InFFrA an interesting extension of the general CS framework with one respect is the fact that agents actually have to strategically decide which utterances to generate in accordance with their current model of the CS to achieve their goals. In [18] we have suggested entropy-based methods for reconciling the utility-based preferences of InFFrA agents with long-term considerations about the effect of their decisions on the overall CS in decision-theoretic terms. There, the interesting question was how agents can achieve a trade-off between their current pursuit for high utility and the modifications to the CS that will result from their current decision.

In [15], we have shown how InFFrA frames can be formally converted to general expectation networks. Of course, some problems occur when attempting to transform general expectation networks to interaction frames, since the full expressiveness of expectation networks is not available in the formal model of InFFrA [17] for reasons of practicability. In future work, we are going to investigate how CS that have been observed by global entities can be used by agents in the system to improve their interaction behaviour.

5. Conclusion

This paper presented communication systems as a unified model for socially intelligent systems based on recording and transforming communicative expectations. We presented formalisms for describing expectations in terms of expectation networks, the formal semantics of these networks, and a general framework for transforming them with incoming observation. Then, a number of important applications of CS were discussed, some of which have already been addressed by our past research, while others are currently being worked on.

While a lot of work still lies ahead, we strongly believe that, by virtue of their general character, CS have the potential of becoming a unified model for speaking about methods and applications relevant to the improvement of multiagent systems using sociological theories [10]. Also, we hope that they can contribute to bringing key insights of this new research direction to the attention of the mainstream DAI audience, as they put emphasis on certain aspects of MAS that are often neglected in traditional approaches.

6. References

1. W. Brauer, M. Nickles, M. Rovatsos, G. Weiß, and K. F. Lorentzen. Expectation-Oriented Analysis and Design. In Procs. AOSE-2001, Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin, 2002.
2. P. R. Cohen and H. J. Levesque. Performatives in a Rationally Based Speech Act Theory. In Procs. 28th Annual Meeting of the ACL, 1990.
3. P. R. Cohen and H. J. Levesque. Communicative actions for artificial agents. In Procs. ICMAS-95, 1995.
4. FIPA, Foundation for Intelligent Agents, http://www.fipa.org.
5. E. Goffman. Frame Analysis: An Essay on the Organisation of Experience. Harper and Row, New York, NY, 1974.
6. F. Guerin and J. Pitt. Denotational Semantics for Agent Communication Languages. In Procs. Agents'01, ACM Press, 2001.
7. Y. Labrou and T. Finin. Semantics and conversations for an agent communication language. In Procs. IJCAI-97, 1997.
8. K. F. Lorentzen and M. Nickles. Ordnung aus Chaos – Prolegomena zu einer Luhmann'schen Modellierung deentropisierender Strukturbildung in Multiagentensystemen. In T. Kron, editor, Luhmann modelliert. Ansätze zur Simulation von Kommunikationssystemen. Leske & Budrich, 2002.
9. N. Luhmann. Social Systems. Stanford University Press, Palo Alto, CA, 1995.
10. Th. Malsch. Naming the Unnamable: Socionics or the Sociological Turn of/to Distributed Artificial Intelligence. Autonomous Agents and Multi-Agent Systems, 4(3):155–186, 2001.
11. G. H. Mead. Mind, Self, and Society. University of Chicago Press, Chicago, IL, 1934.
12. M. Nickles and G. Weiß. Empirical Semantics of Agent Communication in Open Systems. In Procs. 2nd Workshop on Challenges in Open Systems, 2003.
13. M. Nickles and G. Weiß. A framework for the social description of resources in open environments. Procs. 7th International Workshop on Cooperative Information Agents (CIA 2003). Lecture Notes in Computer Science, Volume 2782. Springer-Verlag. 2003.
14. M. Nickles, G. Weiß. Multiagent Systems without Agents – Mirror-Holons for the Compilation and Enactment of Functional Communication Structures. In K. Fischer, M. Florian, editors, Socionics: Its Contributions to the Scalability of Complex Social Systems. Lecture Notes in Artificial Intelligence, Springer Verlag, 2003. To appear.
15. M. Nickles, M. Rovatsos, W. Brauer, G. Weiß. Communication Systems: A Unified Model of Socially Intelligent Systems. In K. Fischer, M. Florian, editors, Socionics: Its Contributions to the Scalability of Complex Social Systems. Lecture Notes in Artificial Intelligence, Springer Verlag, 2003. To appear.
16. J. Pitt and A. Mamdani. A protocol-based semantics for an agent communication language. In Procs. IJCAI-99, 1999.
17. M. Rovatsos and F. Fischer. A Formal Semantics for InFFrA. Research Report FKI-248-03, Department of Informatics, Technical University of Munich, to appear, 2003.
18. M. Rovatsos, M. Nickles, and G. Weiß. Interaction is Meaning: A New Model for Communication in Open Systems. In Procs. AAMAS'03, Melbourne, Australia, to appear, 2003.
19. M. Rovatsos and K. Paetow. On the Organisation of Social Experience: Scaling up Social Cognition. In K. Fischer, M. Florian, editors, Socionics: Its Contributions to the Scalability of Complex Social Systems. Lecture Notes in Artificial Intelligence, Springer Verlag, 2003. To appear.
20. M. Rovatsos, G. Weiß, and M. Wolf. An Approach to the Analysis and Design of Multiagent Systems based on Interaction Frames. In Procs. AAMAS'02, Bologna, Italy, 2002.
21. M. P. Singh. A semantics for speech acts. Annals of Mathematics and Artificial Intelligence, 8(1–2):47–71, 1993.
22. M. P. Singh. A social semantics for agent communication languages. In Procs. IJCAI Workshop on Agent Communication Languages, 2000.