Reasoning about Knowledge and Context-Awareness

Michael Cebulla
Technische Universität Berlin, Fakultät für Elektrotechnik und Informatik
Institut für Softwaretechnik und Theoretische Informatik
mce@cs.tu-berlin.de

Abstract

In this paper we propose a knowledge-based approach for the support of adaptive and context-aware behavior in multi-agent systems. We identify the agents *knowledge* which is present in a system as a central factor for context-awareness and for the systems ability to adapt to changing environments. We provide a framework which supports simple and intuitive specification of complex knowledge configurations. We discuss the semantics and formal backgrounds of the central notion of *knowledge conformance.* Finally we describe an algorithm for reasoning about knowledge conformance, i.e. for deciding whether a knowledge specification holds for an agent (or a group of agents). Our algorithm heavily relies on the features from membrane computing and is especially well-suited for reasoning about incomplete knowledge. In addition it supports the incremental introduction of situational knowledge and thus supports the dynamics of context-awareness.

Introduction

Enabled by the recent advances on the fields of hardware design, wireless communication and (last not least) the Internet the distribution of mobile devices in society is expected to increase dramatically during the forthcoming decade. In the light of these expectations new requirements are established concerning the robustness and reliability of systems. Especially the requirement of context-awareness is critical in pervasive environments: mobile devices have to be aware of information and services which are present in the current situation and cooperate with them when desirable. On the other hand they are also required to provide meaningful behavior when these services cannot be found. In this paper we start with the observation that *knowledge* is an important feature in these applications since it incorporates the information about the environment's current state during runtime.

In this presentation we focus on the systems' ability to represent knowledge about themselves and their situations which is a precondition for reacting adequately to unexpected environmental changes. W.r.t. the dynamics of knowledge our architecture supports two models of *knowledge diffusion: pure* coordination (via uninterpreted indicators) as well as *knowledge-based* coordination (via semantic-based reasoning). We show that the the integration of of these coordination mechanisms is possible on the platform of membrane computing.

	Objective	Method
1	Knowledge specifications	Description Logics
2	Black-Box Reasoning	Observation-based Specification
3	Semantics of Observation and Access	Membrane Computing

Figure 1: Architecture

In this paper we propose to combine ontologies (i.e. description logics, cf. Baader *et al.* 2003) as a light-weight formalism for knowledge representation with membrane computing (Păun 2000) (supporting the aspects of term rewriting and pure coordination) in order to obtain a framework for a knowledge-based high-level modeling of complex systems (as already demanded in Pepper *et al.* 2002).

This paper is organized as follows: first we give an outline of the general architecture. Then we discuss the formal backgrounds of our observation-based approach. After this we describe a membrane-based algorithm for automated reasoning which enables the integration of the two modes of coordination as knowledge diffusion. Finally we discuss some more complex examples in order to demonstrate the possibilities of our approach.

General Architecture

Within the architecture of our proposal we distinguish three layers (cf. Figure 1). While we give support for the high-level specification of complex knowledge on the top level, the basic concepts for black-box simulations and their semantics are provided on the second level. The means for reasoning about these concepts are defined on the bottom level. For the structure of our proposal we are indebted to Pavlovic, Pepper, & Smith 2006.

Syntax of $\mathcal{ALC}_{reg}$. For the creation of knowledge specifications we use the description logic $\mathcal{ALC}_{reg}$ (Calvanese & Giacomo 2003) which extends $\mathcal{ALC}$ (Schmidt-Schauß & Smolka 1991) by complex roles (e.g. regular expressions over roles). The abstract syntax can be described as follows (C, Q, R representing concepts, elementary roles and complex roles respectively):

$$\begin{array}{lcl} C, C' & \rightarrow & \top|\bot|C|C \wedge C'|C \vee C'|\neg C|\forall R.C|\exists R.C \\ R, R' & \rightarrow & Q|R \sqcup R'|R \circ R'|R^*|id(C) \end{array}$$

The semantics of $\mathcal{ALC}_{reg}$ can be defined by the correspondence to the modal logics $\mathbf{K}_m$ (Schild 1991). As we will see in later parts of this paper this enables us to treat description logic formulas as statements about complex configurations of knowledge and to use Kripke-style possible worlds semantics (Kripke 1963).

Black-Box Modeling. Our central goal is to provide a method to decide at runtime whether a given knowledge specification holds for certain agents. Note that these agents may possess much more information and may have numerous additional characteristics. Since these additional features are not relevant for the specific question we ignore them by choosing a style of black-box simulation.

Membrane Computing. We use concepts from membrane computing in order to formulate the algorithms for reasoning about knowledge conformance. Not that these concepts are specifically well-suited for the representation of incomplete knowledge and for the dynamical enrichment with context information. We claim that these characteristics are constitutive for context-aware behavior.

Following (Păun 2000) we use the concept of P-systems which heavily relies on the metaphor of a *chemical solution* (Berry & Boudol 1992) for the representation of knowledge in a system. A solution contains *molecules* which may represent *terms*. As we will see these terms are elements of a specific terminology.

Knowledge Diffusion. Generally we consider two types of coordination mechanism (which can be considered as two extremes in a continuum) which support different aspects of context-awareness. Generally we use the notion of *blackboard*-architecture (Shaw & Garlan 1996) in order to integrate a global environments which holds (syntactic representations of) the entire knowledge which is currently present in the system. By integrating semantic reasoning techniques we support the processing of information which is semantically rich. On the other hand however we have to support very robust coordination mechanisms which do not rely on interpreted messages but on the presence of (uninterpreted) indicators (for the semiotic background cf. Cebulla 1995).

Knowledge-based Integration of Sensor Technology. We claim that our approach is well-suited e. g. for a knowledge-based integration of medical technology. While there have been huge progresses concerning the development of sensors for the measurement of vital data the integration of the devices is still incomplete (Watt, Maslana, & Mylrea 1993). As a consequence a large number of nuisance alarms are bothering the medical staff. A typical case for an undetected interdependence between sensors resulting in false alarms is represented by an alarm related to oxygen saturation when the blood pressure cuff is inflated.

An approach for the knowledge-based integration of the devices and the enabling of context-aware alarms is sketched in Figure 2. The knowledge concerning the fact that the cuff

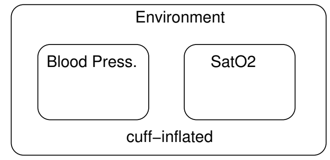

Figure 2: Knowledge-based Integration

for the measuring of the blood pressure is inflated (represented by the molecules *cuff-inflated*) can be used to suppress alarms caused by the signal loss of the pulsoxymetry sensor. This behavior can be described by the following rule.

$$[\mathsf{cuff\text{-}inflated}^{\mathsf{c}}, \mathsf{o2\text{-}alarm}], \mathsf{c} > \tau \rightarrow [\mathsf{cuff\text{-}inflated}^{\mathsf{c}}]$$

Note that this (simplified) rule describes the behavior of the component *Environment* which thus is responsible for the suppression of the alarm. This kind of adaptive behavior can be considered as an example for context-awareness. The molecules *cuff-inflated* is used here in analogy to signal molecules in the coordination of bacteria populations (cf. Krasnogor *et al.* 2005).

Observation-based Specification

Our specific interest in this presentation is directed to the treatment of the question whether an abstract specification of complex knowledge can be met by a specific configuration of agents in a specific environment. In order to deal with this question we use black-box reasoning. In this section we introduce the concepts which we use as foundations for this kind of reasoning. Finally we give a definition for the kind of *knowledge conformance* which we want to prove.

Agents

In our approach we emphasize that in incomplete specifications of agents knowledge their current state (and thus the behavior of the whole systems) may not be known completely. Consequently we opt for an *observation-based* style of knowledge modeling which partly relies on notions from coalgebraic specification (cf. Rutten 1996). Intuitively in this modeling paradigm the agents state space is treated as *unknown* (or infinite) solely supporting temporary observations. As we will see a central issue in our treatment is the accessibility of certain worlds (as known from Kripke semantics) which have to be proved by experiments.

These considerations lead to the simple coalgebraic structure which defines two functions (a value function *obs* and a transition function *acc*) which are defined on the unknown state space X_1.

interface Agent
obs: $X_1 \times T \rightarrow \mathbf{bool}$
next: $X_1 \times T \rightarrow X_1$

Intuitively these functions correspond to a Kripke semantics. While the function *acc* enters an accessible world the function *obs* checks whether certain observations hold in this world. This structure can be considered as an observation interface.

Accessing and Observing

Accessing a World. In this subsection we describe a multiset-based method for the (simplified) specification of accessing a possible world. Intuitively the access to such a world is modeled as a specific kind of *action* in our approach.

Definition 1 (Action) *An action $a = \langle Pre, Post\rangle$ is defined by a name a and by multisets Pre (resp. $Post$) of pre- (resp. post-conditions).*

For the sake of knowledge-based modeling we treat agents as P-systems (Păun 2002). We rely on this central concept from *membrane computing* because this style of modeling explicitly supports the representation of incomplete knowledge (in contrast to traditional styles of automatic reasoning) und thus the description of highly reactive types of behavior. Further we expect it to support the integration of bio-inspired coordination mechanisms into systems modeling (e.g. *quorum sensing,* cf. Krasnogor *et al.* 2005).

Definition 2 (P-System) *A P-system of degree m is defined as a tuple $\Pi = \langle O, \mu, w_1, \dots, w_m, R_1, \dots, R_m\rangle$, where O is an alphabet, μ is a membrane structure, $w_1, \dots, w_m$ are multisets of strings from O, $R_1, \dots, R_n$ are sets of transformation rules associated with the regions.*

For the sake of our simple treatment we can now describe a call of **next**(Ag, acc) in terms of membrane computing. Here *acc* is the name of a specific action whose execution leads to the entrance into a state which is considered as possible by the agent. The membrane which represents this possible world is labeled with this name. In this context we consider the action name acc as a molecule which is introduced into the multiset representing the current state of P-system Ag. This introduction only has an effect if there is a rule $r \in R_{Ag}$ which mentions the reception of molecule acc in its left-hand side.

$$[_{Ag}(acc, in)[_{acc}]_{acc}]_{Ag} \quad \rightarrow \quad [_{acc}]_{acc}$$

Intuitively this rule describes the unpacking of possible worlds. Note that in the standard case there are multiple possible worlds contained in the state of an agent. We silently presume that we can access these states by some index in a tree-like structure.

Observation. For the sake of this presentation we define observation using containment in multisets. An agent's state support an observation when the molecule representing the relevant information is contained in the current region.

Bisimulation

Formal Background. We start with the fact that both the interface *agent* as well as the interface *tree* can be considered as coalgebras. We will see examples for the tree-like structure of knowledge specifications in later parts of this paper).

interface Tree
root: $X_2 \times T \rightarrow \mathbf{bool}$
subtree: $X_2 \times T \rightarrow X_2$

$$\begin{tikzcd} X_1 \arrow[r, "f"] \arrow[d, "\textbf{obs}"'] & X_2 \arrow[d, "\textbf{root}"] \\ T \arrow[r, "id"] & T \end{tikzcd}$$

and

$$\begin{tikzcd} X_1 \times T \arrow[r, "f \times id"] \arrow[d, "\textbf{next}"'] & X_2 \times T \arrow[d, "\textbf{subtree}"] \\ X_1 \arrow[r, "f"] & X_2 \end{tikzcd}$$

Figure 3: Homomorphism on Coalgebras

Similar to the function *obs* there is an observation function *root* which supports observations concerning the value of the tree's root. On the other hand the function *subtree* (corresponding to *next*) supports experiments on the coalgebra. Note that (following Rutten 1998) coalgebras can be considered as deterministic automata $S = \langle S, o, t\rangle$, where S is an infinite state space, o the observation function and t the transition function (corresponding to *next* resp. *subtree*). In addition we say that a state is accepting if the observation function yields *true* in this state.

Consequently we use the notion of *bisimulation* on deterministic automata for reasoning about an agent's conformance w.r.t. a knowledge specification (cf. Rutten 1998).

Definition 3 (Bisimulation) *A bisimulation between two automata $S = \langle S, o, t\rangle$ and $S' = \langle S', o', t'\rangle$ is a relation $R \subseteq S \times S'$ with, for all $s \in S, s' \in S'$, and $a \in A$:*

$$\textit{if } sRs' \textit{ then } \begin{cases} o(s) = o'(s') \textit{ and} \\ t(s)(a)Rt'(s')(a). \end{cases}$$

More technically we have to prove that there exists a homomorphism between the two automata. Such a homomorphism is defined as a function $f : S \rightarrow S'$ which commutes with the functions defined in the interfaces. In our case such a homomorphism could be defined by $obs(s)(a) = root(f(s))(a)$ and by $f(next(s)(a)) = subtree(f(s))(a)$ (cf. Figure 3).

Interestingly enough a bisimulation between automata can be considered as an automaton by itself. Consequently in order to support reasoning about behavioral conformance we have to construct an automaton as follows (cf. Rutten 1998).

Definition 4 (Bisimulation Automaton) *For a bisimulation R between automata S and S' an automaton $R = \langle R, o_R, t_R\rangle$ can be constructed, where $R = S \times S'$, $o_R(\langle s, s'\rangle) = o(s) = o'(s')$ and $t_R(\langle s, s'\rangle)(a) = \langle t(s)(a), t'(s')(a)\rangle$.*

Reasoning about Knowledge

Possible Worlds Semantics. While the formal exploration of mental attitudes was initiated by (Hintikka 1962) the conceptual framework was defined by (Kripke 1963). The notion of Kripke structures is based on the idea that knowledge is not represented by propositions but in terms of *states* in which propositions hold (or do not hold). These states are frequently called *possible worlds*. The basic relation in Kripke structures is the relation of possibility which is modeled by an accessibility relation.

Definition 5 (Kripke Structure) *A* Kripke Structure M *for* n *agents over a modal formalism is a tuple* $\langle W, \pi, R_1, \ldots, R_n \rangle$*, where* W *is a set of possible worlds (or states),* π *is an* interpretation *that associates with each state in* w *a truth assignment to the primitive propositions of modal logics* p *(i.e.,* $\pi(w) : p \rightarrow \{true, false\}$ *for each state* $w \in w$*), and* R_i *is a binary relation on* W.

Note the intuitive correspondence between *obs* and π resp. R and *acc*.

Reasoning. We use the concepts discussed above in order to support reasoning about incomplete knowledge. As an example we consider an agent who has access to several sources of sensor data. Some of these sources may already deliver data while others may not provide usable information. In our treatment each of these data sources is represented as a possible world.

As discussed before we use membrane-based computing for the representation of such possible worlds. These accessible states of a Kripke structure are represented by membranes which are embedded into an agent's local knowledge. These membranes are labeled with the name of the individual accessibility-relation. We use the abbreviation *wl* for *water-level* (which is a variable which is important in scenarios from disaster management).

$$\begin{gathered}[_{Ag_1}[_{acc_1} wl = 0.3]_{acc_1} \\ [_{acc_1} wl = 0.5]_{acc_1} [_{acc_1} wl = 0.6]_{acc_1}]_{Ag_1}\end{gathered}$$

We can now use the bisimulation automaton defined in the previous section in order to check several statements concerning the agent's knowledge. For this sake we transform such statements from description logics expressions into tree-like representations constituted by embedded membranes. Let us first consider statements concerning belief and knowledge.

$$\begin{aligned}\mathsf{K_i water\text{-}level_{\geq 0.3}} &\Leftrightarrow \forall \mathsf{water\text{-}level_{\geq 0.3}} \\ \mathsf{B_i water\text{-}level_{\geq 0.3}} &\Leftrightarrow \exists \mathsf{water\text{-}level_{\geq 0.3}}\end{aligned}$$

We represent these statements by the following membrane expressions.

$$[_T [_\forall [_{acc_1} wl_{\geq} 0.3]_{acc_1}]_\forall]T$$

$$[_T [_\exists [_{acc_1} wl_{\geq} 0.3]_{acc_1}]_\forall]T$$

Membrane-Based Tree-Automaton. Intuitively the automaton checks whether the complex knowledge specification T is a valid observation in the current state of Ag. If this is the case several experiments are performed on the agent (concerning the accessibility of possible worlds). Again these experiments are prescribed by the knowledge specification T. If there are multiple experiments necessary, copies of the automaton are created for every experiment. We exploit the characteristics of membrane computing and its computational properties for the creation and handling of multiple automata.

In our proposal the automaton for bisimulation is implemented as a rewriting P-system.

Definition 6 (P-system for Bisimulation) *A P-system for bisimulation is defined as a tuple* $P_{BS} = \langle V, \mu, w1, \ldots, w_m, R_1, \ldots R_m \rangle$*, where* $V = N \cup T$ *(*N, T *being defined by a suitable grammar) and* $\mu = [_0[_{Ag}]_{Ag}[_T]_T]_0$.

While the membrane $[_{Ag}]_{Ag}$ representing the agent is treated as black box (just supporting calls to *obs* and *next*) $[_T]_T$ contains a tree-like structure of embedded membranes as discussed above.

The P-system defined above corresponds to a tree automaton $\mathbf{A}_{C_0} = \langle \Sigma, S, \delta, s_{ini}, F \rangle$ (as described by (Calvanese & Giacomo 2003)). The following correspondences hold:

- In (Calvanese & Giacomo 2003) the alphabet is defined by $\Sigma = 2^{\mathcal{A}} \times (\mathcal{B} \cup \{\epsilon\})$, i.e. the set of pairs whose first components is a set of atomic concepts and whose second component is a basic role (with basic concepts $\mathcal{A} \subseteq T$, basic roles $\mathcal{B} \subseteq T$ as defined by the terminology). In our reformulation the current input is defined by the tree-operations (*root*, *label*). Note that in our representation these operations remain implicit. While the result of *root* corresponds to the molecules which are contained in the current tree membrane the result of *label* is the label of the current membrane.
- Note that in our approach the state space X_1 of the embedded automaton Ag is treated as unknown. Consequently the state space of the tree automaton is defined by $S = X_1 \times T$.
- The transition relation $\delta : S \times \Sigma \rightarrow \phi$ of the automaton maps a state of the automaton and an input letter to a set of pairs ϕ where each pair (x', t) corresponds to a copy of the automaton (whose state x' is incompletely known) and a subtree t of the knowledge specification. The relation is defined by the following rewriting rules:

  1. Checking whether the root is labeled with ϵ.

     $$[[_{Ag}X]_{Ag}[_T[_\epsilon \alpha]_\epsilon]_T] \quad \rightarrow \quad [[_{Ag}X]_{Ag}[_T C_0]_T]$$

  2. Atomic Concepts $A \in \mathcal{A}$:

     $$\begin{aligned}[[_{Ag}X]_{Ag}[_T[_q A]_q]_T] \quad &\rightarrow \quad [\mathbf{true}],\ \text{when } \mathbf{obs}_{Ag}(A) \\ [[_{Ag}X]_{Ag}[_T[_q A]_q]_T] \quad &\rightarrow \quad [\mathbf{false}],\ \text{when } \neg\mathbf{obs}_{Ag}(A)\end{aligned}$$

     It can be seen that $\mathbf{obs}_{Ag}(A) = A \in X$. Note that these rules describe an observation whether $\mathbf{obs}_{Ag}(\mathbf{root}_T)$. For $\neg A$ the corresponding rules hold.

  3. Basic Roles $Q \in \mathcal{B}$:

     $$\begin{gathered}[[_{Ag}X]_{Ag}[_T[_Q \alpha]_Q]_T] \rightarrow \\ [[_{Ag}\mathbf{next}_{Ag}(X, Q)]_{Ag}[_T \alpha]_T],\end{gathered}$$
     when $\mathbf{next_{Ag}}(Q)$ is enabled.

     $$\begin{gathered}[[_{Ag}X]_{Ag}[_T[_Q \alpha]_Q]_T] \rightarrow \\ [\mathbf{false}],\ \text{when } \mathbf{next}_{Ag}(Q) \text{ is not enabled.}\end{gathered}$$

Note that these rules describe a test whether $\mathbf{next}(q)$ is enabled. Note that in our observation-based approach this can only be tested by a suitable observation, i. e. $obs(next(Ag, label(T)), root(subtree(label(T), T)))$.

4. For the concepts from the behavioral description T the following rules can be defined (Q being again an atomic, R a complex role):

$$[[_{Ag}X]_{Ag}[_T A \sqcap B]_T] \rightarrow$$
$$[[_\sqcap[[_{Ag}X]_{Ag}[_T A]_T][[_{Ag}X]_{Ag}[_T B]_T]]_\sqcap]$$

$$[[_{Ag}X]_{Ag}[_T A \sqcup B]_T] \rightarrow$$
$$[[_\sqcup[[_{Ag}X]_{Ag}[_T A]_T][[_{Ag}X]_{Ag}[_T B]_T]]_\sqcup]$$

$$[[_{Ag}X]_{Ag}[_T \forall Q.C]_T] \rightarrow$$
$$[[_\sqcap[[_{Ag}\mathbf{next}^i]_{Ag}[_T \neg Q]_T][[_{Ag}\mathbf{next}^i]_{Ag}[_T C]_T]]_\sqcap],$$
for all $i \in [1, k_T]$

$$[[_{Ag}X]_{Ag}[_T \exists Q.C]_T] \rightarrow$$
$$[[_\sqcup[[_{Ag}\mathbf{next}^i]_{Ag}[_T Q]_T][[_{Ag}\mathbf{next}^i]_{Ag}[_T C]_T]]_\sqcup]_\sqcup],$$
for all $i \in [1, k_T]$

$$[[_{Ag}X]_{Ag}[_T \forall R^*.C]_T] \rightarrow$$
$$[[_\sqcap[[_{Ag}X]_{Ag}[_T C]_T][[_{Ag}X]_{Ag}[_T \forall R.\forall R^*.C]_T]]_\sqcap]]$$

$$[[_{Ag}X]_{Ag}[_T \exists R^*.C]_T] \rightarrow$$
$$[[_\sqcup[[_{Ag}X]_{Ag}[_T C]_T][[_{Ag}X]_{Ag}[_T \exists R.\exists R^*.C]_T]]_\sqcup]$$

Note that we omit the rules for the treatment of most role constructors in this description. In the rules for the processing of quantified role expressions (involving atomic roles) the terms labeled with i have to be created for each $i \in [1, k_T]$ where k_T is the number of successor states in the current tree. Intuitively this corresponds to the creation of a copy for each role filler of Q.

An example for the application of rules is given in Figure 4. The state of the global automaton is represented by the membrane labeled with 0. The application of the rule for the treatment of value restrictions is shown. A copy of the automaton is created for each of three rule fillers (representing possible worlds) which are present in the local knowledge of Ag_1. In the next step each automaton checks if the constraint ≥ 0.3 holds. If each automaton comes to a positive conclusion the initial statement concerning the knowledge $K_1 water\text{-}level_{\geq 0.3}$ holds.

Acceptance. The knowledge specification T is accepted by the automaton Ag if there is an accepting run of the global automaton. For the general case of infinite knowledge specifications we have to formulate a Büchi-criterion which requires that accepting states do not occur infinitely often during a run.

Complex Specifications of Knowledge

In the final section of this paper we briefly discuss some additional features in order to demonstrate the possibilities of our proposal. Especially we extend our notions in order to support the reasoning about the knowledge in groups of

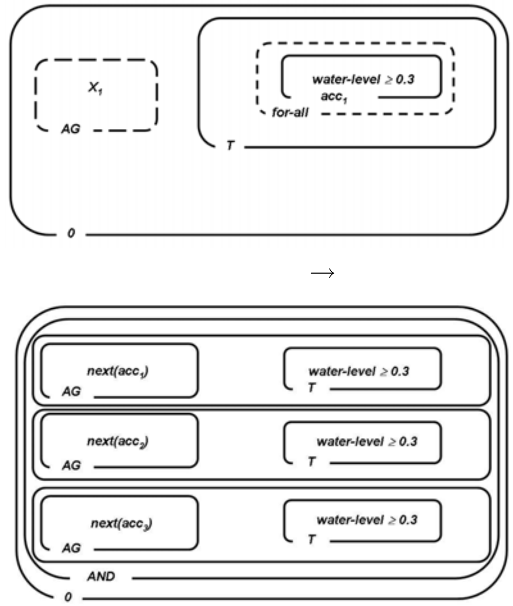

Figure 4: Rule Application

agents. Following (Fagin *et al.* 1996) again we define shared knowledge as conjunction of individual knowledge.

$$E_G\phi \Leftrightarrow \bigwedge_{i \in G} \mathbf{K}_i\phi$$

Intuitively in our membrane-based calculus the employment of a predicate like E_G would have the consequence that three copies of the automaton were created which each check whether the individual knowledge of each of the three individual agents conforms to the (relevant parts of the) specification. Consequently in order to describe the integration of shared knowledge into our calculus we can use the following rule (formulated for three agents).

$$[[_G X]_G[_T E_G A]_T] \rightarrow$$
$$[[_\sqcap[[_1 X]_1[_T \forall acc_1.A]_T]$$
$$[[_2 X]_2[_T \forall acc_2.A]_T]$$
$$[[_3 X]_3[_T \forall acc_3.A]_T]]_\sqcap]$$

Common Knowledge. A little bit more complicated (but very similar) is the treatment of common knowledge. Common knowledge which is frequently considered as a necessary precondition for coordination in multi-agent systems (Fagin *et al.* 1996) can be intuitively defined as recursive shared knowledge concerning certain facts. This intuition is captured by the following fixpoint equation which has many interesting properties. Common knowledge is defined in the following way:

$$C_G\phi \Leftrightarrow E_G(\phi \wedge C_G\phi).$$

Although this definition is quite ambitious and leads to infinite specifications (when applied consequently) it can be processed easily by our computational framework based on multiset transformation. Although the strict reasoning about this formula could indeed lead to an infinite computation in our approach the result or each reasoning step is a multiset representing incomplete knowledge. Since we explicitly support the processing of incomplete knowledge we can rely on our current knowledge (represented by the multiset) and proceed with the next steps of reasoning at a later time. The treatment of common knowledge can be described by the following rule.

$$
\begin{array}{l}
[[_G X]_G [_T C_G A]_T] \rightarrow \\
[[_\sqcap [[_1 X]_1 [_T \forall acc_1.(A \sqcap C_G A)]_T] \\
[[_2 X]_2 [_T \forall acc_2.(A \sqcap C_G A)]_T] \\
[[_3 X]_3 [_T \forall acc_3.(A \sqcap C_G A)]_T]]_\sqcap]
\end{array}
$$

Conclusions

In this paper we focused on the diffusion of knowledge in complex system as a major factor shaping global behavior. We described a formal approach for representing and processing knowledge specifications in dynamic environments. Especially we proposed an automated procedure for the decision whether a given specification holds for a given configuration of agents. Such reasoning enables a better understanding of systems behavior during runtime. Especially it can be analyzed if the current distribution of knowledge (among the agents) in the current situation conforms to the specification.

We argued that a sensitive handling of knowledge diffusion enables new possibilities concerning the understanding and creation of novel kinds of behavior. Especially the transfer of sophisticated interactions and coordination mechanisms from fields like biology or sociology seems to be promising on the basis of this paradigm. As direct benefits of such an approach we emphasize increased abilities to provide meaningful behaviors in dynamic environments and pervasive setting (i.e. context awareness).

Acknowledgments

We are indebted to the anonymous reviewers for their helpful comments on an earlier version of this paper.

References

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds. 2003. *The Description Logic Handbook*. Cambridge, U.K.: Cambridge University Press.

Berry, G., and Boudol, G. 1992. The chemical abstract machine. *Journal of Theoretical Computer Science* 96(1):217–248.

Calvanese, D., and Giacomo, G. D. 2003. Expressive description logics. In Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds., *The Description Logic Handbook*. Cambridge University Press, Cambridge, U.K.

Cebulla, M. 1995. Pragmatik und KI: zur semiotischen Fundierung der Sprechakttheorie (in german). *S-European Journal for Semiotics* 7(1,2).

Fagin, R.; Halpern, J. Y.; Moses, Y.; and Vardi, M. Y. 1996. *Reasoning about Knowledge*. Cambridge, Mass.: The MIT Press.

Hintikka, J. 1962. *Knowledge and Belief: An Introduction to the Logic of the two Notions*. Ithaca, New York: Cornell Univ. Press.

Krasnogor, N.; Georghe, M.; Terrazas, G.; Diggle, S.; Williams, P.; and Camara, M. 2005. An appealing computational mechanism drawn from bacterial quorum sensing. *Bulletin of the European Association for Theoretical Computer Science*.

Kripke, S. 1963. A semantic analysis for modal logic i: normal modal propositional calculi. *Zeitschrift für Mathematische Logik and Grundlagen der Mathematik* 9:67–96.

Păun, G. 2000. Introduction to membrane computing. *Journal of Computer and System Science* 61(1):108–143.

Păun, G. 2002. *Membrane Computing. An Introduction.* Berlin: Springer.

Pavlovic, D.; Pepper, P.; and Smith, D. 2006. Specification engineering. *to appear*.

Pepper, P.; Cebulla, M.; Didrich, K.; and Grieskamp, W. 2002. From program languages to software languages. *The Journal of Systems and Software* 60.

Rutten, J. 1996. Universal coalgebra: a theory of systems. Technical report, Centrum voor Wiskunde en Informatica, Amsterdam.

Rutten, J. 1998. Automata and coinduction (an exercise in coalgebra). Technical report, Centrum voor Wiskunde en Informatica, Amsterdam.

Schild, K. 1991. A correspondence theory for terminological logics: Preliminary report. Technical Report KIT-Report 91, Technische Universität Berlin, Franklinstr. 28/29, Berlin, Germany.

Schmidt-Schauß, M., and Smolka, G. 1991. Attributive concept descriptions with complements. *Artificial Intelligence* 48(1):1– 26.

Shaw, M., and Garlan, D. 1996. *Software Architecture. Perspectives on an emerging discipline*. Upper Saddle River, N.J.: Prentice Hall.

Watt, R. C.; Maslana, E. S.; and Mylrea, K. C. 1993. Alarms and anesthesia. challenges in the design of intelligent systems for patient monitoring. *IEEE Engineering in Medicine and Biology* 12(4):34 – 41.