Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Research Report
RR-99-03

Holonic Multi-Agent Systems

Christian Gerber, Jörg Siekmann, Gero Vierke

May 1999

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
67608 Kaiserslautern, FRG
Tel.: + 49 (631) 205-3211
Fax: + 49 (631) 205-3210
E-Mail: info@dfki.uni-kl.de

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel.: + 49 (681) 302-5252
Fax: + 49 (681) 302-5341
E-Mail: info@dfki.de

WWW: http://www.dfki.de

Deutsches Forschungszentrum für Künstliche Intelligenz

DFKI GmbH

German Research Center for Artificial Intelligence

Founded in 1988, DFKI today is one of the largest nonprofit contract research institutes in the field of innovative software technology based on Artificial Intelligence (AI) methods. DFKI is focusing on the complete cycle of innovation — from world-class basic research and technology development through leading-edge demonstrators and prototypes to product functions and commercialization.

Based in Kaiserslautern and Saarbrücken, the German Research Center for Artificial Intelligence ranks among the important “Centers of Excellence” worldwide.

An important element of DFKI's mission is to move innovations as quickly as possible from the lab into the marketplace. Only by maintaining research projects at the forefront of science can DFKI have the strength to meet its technology transfer goals.

DFKI has about 115 full-time employees, including 95 research scientists with advanced degrees. There are also around 120 part-time research assistants.

Revenues for DFKI were about 24 million DM in 1997, half from government contract work and half from commercial clients. The annual increase in contracts from commercial clients was greater than 37% during the last three years.

At DFKI, all work is organized in the form of clearly focused research or development projects with planned deliverables, various milestones, and a duration from several months up to three years.

DFKI benefits from interaction with the faculty of the Universities of Saarbrücken and Kaiserslautern and in turn provides opportunities for research and Ph.D. thesis supervision to students from these universities, which have an outstanding reputation in Computer Science.

The key directors of DFKI are Prof. Wolfgang Wahlster (CEO) and Dr. Walter Olthoff (CFO).

DFKI's six research departments are directed by internationally recognized research scientists:

- Information Management and Document Analysis (Director: Prof. A. Dengel)
- Intelligent Visualization and Simulation Systems (Director: Prof. H. Hagen)
- Deduction and Multiagent Systems (Director: Prof. J. Siekmann)
- Programming Systems (Director: Prof. G. Smolka)
- Language Technology (Director: Prof. H. Uszkoreit)
- Intelligent User Interfaces (Director: Prof. W. Wahlster)

In this series, DFKI publishes research reports, technical memos, documents (eg. workshop proceedings), and final project reports. The aim is to make new results, ideas, and software available as quickly as possible.

Prof. Wolfgang Wahlster
Director

Holonic Multi-Agent Systems

Christian Gerber, Jörg Siekmann, Gero Vierke

DFKI-RR-99-03

This work has been supported by Siemens AG and The Federal Ministry of Education, Science, Research, and Technology (FKZ ITW-95004).

ISSN 0946-008X

Holonic Multi-Agent Systems

Christian Gerber, Jörg Siekmann, Gero Vierke

May 12, 1999

Abstract

A *holonic multi-agent* paradigm is proposed, where agents give up parts of their autonomy and merge into a "super-agent" (a holon), that acts — when seen from the outside — just as a single agent again.

We explore the spectrum of this new paradigm, ranging from definitorial issues over classification of possible application domains, an algebraic characterization of the merge operation, to implementational aspects: We propose algorithms for holon formation and on-line re-configuration. Based on some general criteria for the distinction between holonic and non-holonic domains, we examine domains suitable for holonic agents and sketch the implementation of holonic agents in these scenarios. Finally, a case study of a holonic agent system is presented in detail: TeleTruck system is a fleet management system in the transportation domain.

Contents

1 Introduction

A multi-agent system (MAS) consists of a collection of individual agents, each of which displays a certain amount of *autonomy* with respect to its actions and perception of a domain. Overall computation is achieved by the *autonomous computation* within each agent and by *communication among the agents*. The capability of the whole MAS is an *emergent functionality* that surpasses some of the capabilities of each of the individual agents (see for example [MSR99, SRW98, MWJ97] for recent monographs on multi-agent systems).

The field of MAS is part of distributed AI in the sense that a MAS lends itself naturally to distributed problem solving, where each agent has the characteristics of a distinct problem solver for a specific task. In a complex domain this is an extremely useful feature for the designer of a MAS, as the overall task can now be broken down into a variety of specific subtasks, each of which can be solved by a specific problem solver which can be agentified.

Many distributed problems exhibit a recursive structure: an agent that solves the overall problem may have a similar structure as the agents for the sub-problems, thus they should be structured recursively. More generally, an agent that appears as a single entity to the outside world may in fact be composed of many sub-agents and conversely, many sub-agents my decide that it is advantageous to join into the coherent structure of a super-agent and thus act as single entity — just as the swarm of a certain species of fish sometimes takes on the appearance of a (much bigger) fish. We call agents consisting of sub-agents with the same inherent structure *holonic agents*.

According to Arthur Koestler [Koe67], a *holon* is a natural or artificial structure that is stable and coherent and that consists of several holons as sub-structures. Koestler gives biological examples, for instance a human being consists of organs which in turn consist of cells that can be further decomposed and so on. None of these components can be understood without its sub-components or without the super-component it is part of.

We shall elaborate several approaches on how to design and implement holonic multi-agent systems, using current definitions of agency as a starting point.

The paper is structured as follows: In the next section, we shall compare several agent definitions from sociology and AI. Sketching Koestler's original concept of a holonic organization we shall show how a technical realization can be obtained for the field of holonic manufacturing systems. In Section 3, we use the agent definition of the previous section as a basis for a definition of holonic agents and present a mathematical characterization of holonic agents. Section 4 is concerned with the central issue of resource management in holonic agent systems and gives a categorization of different resource allocation mechanisms for holonic systems. In Section 5, we investigate the suitability of our holonic paradigm for different application domains and in Section 6, we propose a general implementation framework for holonic agent systems. We demonstrate the applicability of this

work in Section 7 using a large fleet management system that is implemented as a holonic MAS.

2 Basic Principles

Autonomy is not only a vital property of classic agent systems, but also a critical issue for holonic systems where agents joining a super-agent have to surrender (some of) their autonomy. Hence, we shall present a collection of well-known definitions for agenthood using them as a starting point for our characterization of holonic agents.

Models of Agency and Autonomy

The great variety of agent or actor definitions ranges from philosophically and sociologically inspired concepts to logical definitions and to definitions that focus on implementational aspects such as a software architecture, efficiency, or tractability.

The sociologist Parsons [Par69] takes an *actor* to be an agent who has goals. In his definition, an *agent* is an individual who shows behavior. *Behavior* is the ability to change the state of the world. The *world* is differentiated into the agent itself and its environment. The *environment*, as it is perceived by the agent, defines the situation the agent is in. A *goal* is a certain state of the world. *To act* means to behave in such a way as to achieve a goal. In general, an agent can choose from a set of actions, about of he has certain expectations how they will change the world. The actor selects a specific action from his options according to his goals, the means at his disposal and his situation. Additionally, agents can use a common language in order to communicate with other actors.

Bratman [Bra87], based on his analysis of rational human behavior on three mental categories *belief*, *desire* and *intention*, connected these by postulating certain requirements for an intelligent agent's mental capabilities. Based on these concepts, Cohen and Levesque [CL90] and Rao and Georgeff [RG91] founded a logical theory of belief, desire and intention (nowadays often called BDI theory) which ascribes these mentalistic notions to artificial agents as well.

Shoham [Sho91] characterizes the term agent as "an entity whose state is viewed as consisting of mental components such as beliefs, capabilities, choices and commitments". He proposes a computational framework for *agent oriented programming* that extends the object oriented programming paradigm by these "mentalistic" notions.

Russell and Norvig [RN95] define an agent as "anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors."

Lange [Lan98] provides a more pragmatic definition that is oriented towards industrial demands: He defines an agent as a software object that has the following properties: situatedness, reactivity, autonomy with respect to its actions, and pro-activity. Furthermore, an agent should be continuously executing. Optionally, an agent can be communicative, mobile, believable or able to learn.
Wooldridge and Jennings [WJ95] characterize an agent by the following traits:

- **Autonomy** with respect to states and actions in the sense of Castelfranchi [Cas95],
- **Social ability**: Agents communicate with other agents via a common language,
- **Reactivity**: Agents respond to changes in their environment which they can perceive, and
- **Pro-activeness**: Agents display a goal-directed behavior based on deliberation in addition to their direct reaction to the environment.

2.1 Requirements for Agency

All these (and other) definitions lay different emphasis on agency, since their requirements have been derived for different purposes. For instance, some definitions require explicitly represented mental states like intentions or beliefs while other definitions admit an implicit representation of these notions. For our purpose, we shall focus on the following characteristics:

- **Autonomy** According to Castelfranchi[Cas95], agent autonomy means that "agents control their actions and internal states to enable them to operate without the direct intervention of humans or others." Russell and Norwig [RN95] define an agent's behavior "autonomous to the extent that its behavior is determined by its own experience." We decompose autonomy into three aspects:
  - **State Autonomy** An agent's state is determined only by its previous states and its perception.
  - **Action Autonomy** Like in Castelfranchi's definition, the action of an agent is determined solely by its current state.
  - **Computational Autonomy** The agent either has computational means of its own or is supplied with computation time (and space) in a fair manner.
- **Goal-Directed Behavior** An agent has explicitly or implicitly represented goals and desires, where desires are defined as in BDI-theory.

- **Action** is defined as in the requirements of Wooldridge and Jennings for reactivity and pro-activeness.
- **Belief** Agents have implicit and/or explicit representations of their environment.
- **Bounded Rationality** In analogy to Russell and Wefald [RW91], we require a rational agent to behave optimally with respect to its limited resources and its goals.
- **Communication** Similar to Wooldridge and Jennings' requirement of social ability, the agents share a communication language.

We shall now extend these requirements to a definition of holonic agents, but first let us introduce the term holon.

2.2 The Holonic Principle

The term *holon*, a combination of the Greek "holos" (whole) and the suffix "-on" (part), was originally introduced in 1967 by the Hungarian philosopher Arthur Koestler [Koe67] in order to name recursive and *self-similar* structures in biological and sociological entities. According to Koestler a holon is a biological or sociological structure that is stable and coherent and that consists of further holons that function similarly. No natural structure is either "whole" or "part" in an absolute sense, instead every holon is a composition of subordinate parts as well as part of a larger whole. For example, a human individual is on the one hand a composition of organs consisting of cells that can be further decomposed, and on the other hand he or she may be part of a group which in turn is part of the human society.
The organizational structure of a holonic society, or *holarchy*, offers advantages that the monolithic design of most technical artifacts lack: They are robust in the face of external and internal disturbancesand damage, they are efficient in their use of resources, and they can adapt to environmental changes.
Koestler's ideas have been applied inter alia in Flexible Manufacturing Systems (see e.g. [Dee94]), where the positive features of Koestler's holarchies, namely stability, adaptability, flexibility, and efficiency motivated a similar design for sufficiently redundant manufacturing processes.[1]
Holons in Holonic Manufacturing Systems (HMS) are characterized by their *holonic attributes*, namely autonomy and cooperativeness. Here, autonomy has roughly the same semantics as for multi-agent systems; namely, the ability to create and control the execution of plans and strategies. Cooperativeness stands for joint planning and coordination of joint plan execution, and, therefore, is

[1] see e.g. the HMS web page `http://hms.ifw.uni-hannover.de/public/hms_tech.html`

subsumed by our agent's attribute of social ability. In a holonic manufacturing system, holons consist of an information processing part and sometimes a physical part, which is responsible for transforming, transporting, storing and validating information as well as physical objects. Manufacturing holons can be build recursively out of other holons.
The idea of agents which in turn consist of agents is not new to the AI community either: for example Minsky's *Society of Mind* [Min86] from 1986 proposed that the human mind is structured as a well organized society of actors/agents.

3 Holonic Agents

We now define a holonic multi-agent system based on this principle, where we use the terms *holon* and *holonic agent* synonymously. By *super-holon* we denote a composition of subordinate agents, which we shall call *sub-holons* or *sub-agents*. As these sub-holons may be further decomposable into sub-sub-holons we shall use the term *immediate sub-holons* to distinguish it from its transitive closure.

3.1 Informal Definition

Arbitrary structures can be viewed as holons in Koestler's framework, where the sub-structures do not necessarily have to be of the same kind. In contrast we like to restrict all entities to agents as defined above, and furthermore, we require that sub-holons always have the same structure as the super-holon. This requirement may later turn out to be too restrictive when the field of holonic MAS matures; as for the moment this restriction makes it easier to define the merge of agents. Now, the essential idea is as follows: A holonic agent of a well-defined software architecture may join several other holonic agents to form a super-holon; this group of agents now acts *as if it were a single holonic agent* with the same software architecture. The nature of the merge of several separate entities into one entity is the subject of the next section, but let us first recall our basic criteria for agency again and extend them to holonic agents:

Autonomy Several agents forming a holon act as a single entity. The holon interacts with the environment as an autonomous agent in the sense of the above presented criteria (state autonomy—action autonomy—computational autonomy).
By joining a holon, agents accept some restriction of their autonomy: they commit to the goals of the holon and they accept restrictions of their abilities to act and to communicate (a detailed description is given in Section 4.2). Nevertheless, they may keep their autonomy to some extend; in particular, they are free to leave the holon.

Common goal-directed behavior Sub-agents of a holon still pursue their private goals and by doing so, they have to pursue at least one common goal of the super-holon which may be represented explicitly or implicitly.[2] Hence, the super-holon's overall goals emerge from the common goals of the sub-agents. We do not require that the super-holon's goals are also the goals of its sub-agents, but the goals of the super-holon and its those of sub-agents must not contradict. Consequently, an agent can only be a member of several holons with conflicting goals if the agent is indifferent to these goals. This requirement corresponds well to the cooperation feature of an HMS holon.

Increased group capabilities An agent's capabilities to act are extended at the group level to *macro* actions which are composed of the actions of the sub-holons. hence, a super-holon may have actions at its disposal that none of its sub-agents could perform alone.

Belief The requirements for an agent's belief remain unchanged: holons have some representation of their environment, i.e. they hold beliefs about their surroundings. This knowledge might be represented explicitly within the super-holon, or it may be distributed and implicitly provided by the local knowledge of the individual sub-holons. Inconsistency between the holon's and some of its members' beliefs is of course an issue and there are several remedies, for example to allow para-consistencies.

Bounded rationality A holon has to control its resources in order to exhibit a bounded rational behavior. Resource management of the sub-holons is monitored by the super-holon, which distributes guidelines to its sub-holons for local resource management. This is an essential issue of our holon definition and will be discussed further in Section 4.2.

Communication The ability to communicate is an essential part of an agent's autonomy. In our framework, we require that the right to communicate with other agents is an exclusive resource of the holon and not of its sub-holons. This right corresponds to a communication channel between two equally ranked holons. Such channels are managed solely by the super-holon.
However, we have to distinguish between communication inside the super-holon and communication between super-holons: internal communication among the sub-holons is of course allowed, but still, even the right to communicate inside the holon is controlled by the super-holon. Efficient holonic resource management is discussed in more detail in Section 4.2. Since problem solving inside a holon

[2]For example, BDI architectures provide an explicit representation of goals. Implicit goals can be ascribed to any agent that exhibits some kind of pro-activeness as defined in Section 2.

is cooperative, internal communication load can be high, hence, an efficient data structure for internal communication should be provided.

3.2 Formal Definition

We define a MAS and multi-agent environments (MAE) in a uniform state based mathematical model in order to reason about holonic structure. This unification is achieved by defining equivalent representations of the system that allow us to freely move between the states of the agents and the state of the environment. We show that any environment containing multiple agents can be isomorphically mapped to an environment in which only one agent is represented explicitly, while the others are integrated into the world. Secondly, we show how to construct for a given MAE an isomorphic one where a group of agents is merged into one *holonic agent*. Vice versa, we speak of a *holonic decomposition* if an agent is decomposed into a group of holonic agents. Later, in Section 4.2.2 we add the concepts of resources and utility to our model.

Definition 1 *A* multi-agent environment *(MAE) is a tuple* $(\mathcal{A}, \mathcal{E}, \Pi, \Delta)$*, where* $\mathcal{A} = \{\alpha_1, \ldots, \alpha_n\}$ *is the set of all agents. Each agent* α_i *is a tuple* (S_i, P_i, A_i, ϕ_i) *of the set of its possible states* S_i*, the sets of percepts* P_i *and actions* A_i*, and its agent function* $\phi_i : S_i \times P_i \to S_i \times A_i$*.* $\mathcal{E}$ *is the set of environmental states.* $\Pi : \mathcal{E} \to (P_1 \times \ldots \times P_n)$ *is a perception function and* $\Delta : \mathcal{E} \times (A_1 \times \ldots \times A_n) \to \mathcal{E}$ *is an environment function.*

In this definition we assume a discrete time scale as in the situation calculus, where a time step is given by the transition from one point of the time scale to the next. For all states of the environment $e \in \mathcal{E}$ and all agent states $(s_1, \ldots, s_n) \in S_1 \times \ldots \times S_n$, each agent α_i receives its local percept $\Pi^i(e)$ via the perception function during each step. The agent computes its action $a_i = \phi_i^2(s_i, \Pi^i(e))$ and its new state $s_i' = \phi_i^1(s_i, \Pi^i(e))$ from its current state s_i and from this perception[3]. The state of the world changes with the actions of the agents:

$$e' = \Delta(e, a_1, \ldots, a_n)$$

denotes the successor environmental state, and

$$s_i' = \phi_i^1(s_i, \Pi^i(e))$$

gives the successor states of the agents for all i. The state transition function $\bar{\Delta} : \mathcal{E} \times S_1 \times \ldots \times S_n \to \mathcal{E} \times S_1 \times \ldots \times S_n$, which is defined as $\bar{\Delta}(e, s_1, \ldots, s_n) = (e', s_1', \ldots, s_n')$, unifies states of the agents and the states of the environment into world states. Hence, the perception function, the agent function and the

[3]An upper index denotes Kleene's projection function, for example $\Pi^i(e)$ is P_i in the codomain $(P_1, \ldots, P_n)$ of Π.

environment function are part of the world transition function $\bar{\Delta}$. The implication of the frame problem for such a state based approach and other subtilities of formalism from the field of cognitive robotics and multi-agent systems are outside of the scope of this section. Our sole purpose here is to provide a formalism that allows us to freely switch from an environmental state to an agent state with a minimum of assumptions (just a state based approach) in order to reason about the holonic merge or decomposition.
Integrating the state of the environment and agents to world states enables us to observe the same multi-agent setting from different perspectives: We can represent any entity in the world explicitly as an agent or implicitly as a part of the environment.

Definition 2 *Two multi-agent environments* $(\mathcal{A}, \mathcal{E}, \Pi, \Delta)$ *and* $(\mathcal{A}', \mathcal{E}', \Pi', \Delta')$ *are* isomorphic *if there exists a bijection function* $\Psi : \mathcal{E} \times S_1 \times \ldots \times S_n \to \mathcal{E}' \times S'_1 \times \ldots \times S'_m$ *such that for all* $(e, s_1, \ldots, s_n) \in \mathcal{E} \times S_1 \times \ldots \times S_n$

$$\bar{\Delta}'(\Psi(e, s_1, \ldots, s_n)) = \Psi(\bar{\Delta}(e, s_1, \ldots, s_n))$$

Notice that for any MAE there exist two special cases: first, an environment without any agents where all state transition is encoded into the environment function Δ and second, a MAE containing only one constant environmental state and one agent where all state transition is encoded into the agent function ϕ. This formalization allows us to reduce a MAE containing several agents to the single agent case by representing all agents but one as entities of the environment. Figure 1 shows the intuition behind this construction, which is explicitly stated in the following lemma.

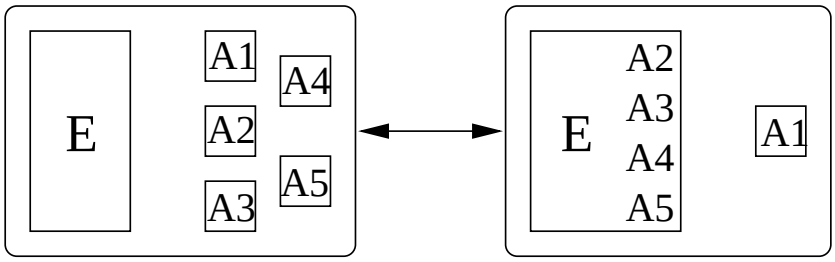

Figure 1: Merging a multi-agent environment into a single-agent environment

Lemma 1 *If* $E = (\{\alpha_1, \ldots, \alpha_n\}, \mathcal{E}, \Pi, \Delta)$ *is a multi-agent environment then for each* $i \leq n$ *there exists an isomorphic multi-agent environment* $(\{\alpha_i\}, \mathcal{E}', \Pi', \Delta')$.

Proof: Without loss of generality, let $i = 1$. We construct an environment $E' = (\{\alpha_1\}, \mathcal{E}', \Pi', \Delta')$ such that $\mathcal{E}' = \mathcal{E} \times S_2 \times \ldots \times S_n$, $\Pi' = \Pi^1$, and for

all $(e, s_2, \ldots, s_n) \in \mathcal{E}'$ and $a_1 \in A_1$ the world function $\Delta' : \mathcal{E}' \times A_1 \to \mathcal{E}'$ is defined as

$$\Delta'((e, s_2, \ldots, s_n), a_1) = (e', s_2', \ldots, s_n')$$

where $e' = \Delta(e, a_1, \phi_2^2(s_2, \Pi^2(e)), \ldots, \phi_n^2(s_n, \Pi^n(e)))$ is the successor state of e in E with respect to a_1 and the other agents' actions $\phi_i^2(s_i, \Pi^i(e))$. $s_i' = \phi_i^1(s_i, \Pi^i(e))$ is the successor state of s_i in E for $2 \leq i \leq n$. The property of isomorphism follows directly from the construction. ■

Let us now introduce the notions of a holonic merge and decomposition and show how to merge a set of agents into a holon, as shown in Figure 2.

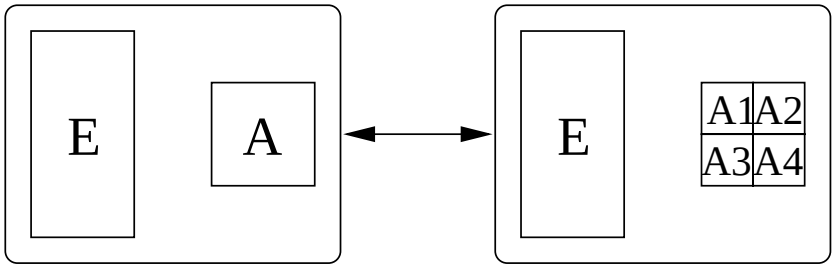

Figure 2: The holonic merge

Definition 3 *Consider the two isomorphic MAE* $(\{\alpha_1, \ldots, \alpha_n\}, \mathcal{E}, \Pi, \Delta)$ *and* $(\{\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i,1}, \ldots, \alpha_{i,m}, \alpha_{i+1}, \ldots, \alpha_n\}, \mathcal{E}, \Pi', \Delta')$. *We call* α_i *the* holonic merge *of* $(\alpha_{i,1}, \ldots, \alpha_{i,m})$.

Lemma 2 *For every MAE* $(\{\alpha_1, \ldots, \alpha_n\}, \mathcal{E}, \Pi, \Delta)$ *and* $k \leq n$ *we can construct an isomorphic multi-agent environment* $(\{\alpha', \alpha_{k+1}, \ldots, \alpha_n\}, \mathcal{E}, \Pi', \Delta')$, *where* α' *is a holonic merge of* $\alpha_1, \ldots, \alpha_k$.

Proof: We construct an environment $(\{\alpha', \alpha_{k+1}, \ldots, \alpha_n\}, \mathcal{E}, \Pi', \Delta')$ with an agent $\alpha' = (S', P', A', \phi')$ that emulates the agents $\alpha_1, \ldots, \alpha_k$ and we adapt the perception function and environment function accordingly: $S' = (S_1 \times \ldots \times S_k)$, let $P' = (P_1 \times \ldots \times P_k)$, $A' = (A_1 \times \ldots \times A_k)$, and for all $s' = (s_1, \ldots, s_k) \in S', p' = (p_1, \ldots, p_k) \in P'$, the agent function is defined as

$$\phi'(s', p') = ((\phi_1^1(s_1, p_1), \ldots, \phi_k^1(s_k, p_k)), (\phi_1^2(p_1, s_1), \ldots, \phi_k^2(p_k, s_k)))$$

where the first(second) arity of ϕ' is a tuple of the first(second) arities of the composed agents' functions. Furthermore, we define the perception function $\Pi' : \mathcal{E} \to (P', P_{k+1}, \ldots, P_n)$ by

$$\Pi'(e) = ((\Pi^1(e), \ldots, \Pi^k(e)), \Pi^{k+1}(e), \ldots, \Pi^n(e))$$

for all $e \in \mathcal{E}$. Here, the perception of α' is composed from the perception of $\alpha_1 \ldots \alpha_k$. Finally, the environment function $\Delta' : \mathcal{E} \times A' \times A_{k+1} \times \ldots \times A_n$ is defined as

$$\Delta'(e, a', a_{k+1}, \ldots, a_n) = \Delta(e, a_1, \ldots, a_n)$$

for all $e \in \mathcal{E}$, $a' = (a_1, \ldots, a_k) \in A'$, $a_{k+1} \in A_{k+1}, \ldots, a_n \in A_n$. Isomorphism follows directly from the construction. ■

Lemma 1 and Lemma 2 enable us to view a collection of agents as one super-holon and to reduce the state transition of the super-holon to the single-agent case.

3.3 Algebraic Properties of the Holonic Merge Operation

The general idea of this paper, namely that several sub-holons can *merge* into one super-holon that displays the same structure to the outside world as the sub-holons, it is composed of, can be seen as a special form of functional composition

$$(H_1 \otimes H_2)$$

where each holon H_1 and H_2 computes some agent function $\phi_i : S_i \times P_i \rightarrow S_i \times A_i, i = 1, 2$. Now the merge has some interesting properties which constitute an Abelian Monoid.

Definition 4 *Let I be the idle holon, which computes the identity function, i.e.*

$$\forall \quad H \; : I \otimes H = H \otimes I = H$$

Lemma 3 *The merge is associative, i.e. for all H_1, H_2, H_3*

$$(H_1 \otimes H_2) \otimes H3 = H_1 \otimes (H_2 \otimes H3)$$

Proof: Follows immediately from the state definitions and the fact that the crossproduct, which gives the semantics is associative by definition. ■

Corrollary 1 *$(H_i, I, \otimes)$ is a monoid.*

Lemma 4 *The merge is commutative, i.e.*

$$H_1 \otimes H_2 = H_2 \otimes H_1$$

The following lemma follows also easily from our definition of a state as the tuple (s, s) is not the same as the single state s.[4]

[4]Of course we could have chosen a formalism for the semantics where tuples of identical states collapse into a singleton, however this would violate our intention: two (twin) agents can achieve more than a single one and this becomes particularly apparent when we take resources into account.

Lemma 5 *The merge is not idempotent, i.e.*

$$H \otimes H \neq H$$

Corrollary 2 $(H_i, I, \otimes)$ *is an Abelian Monoid. It is not idempotent.*

Abelian Monoids are a rather natural algebraic structure desirable in many subfields of theoretical computer science. Special holonic agents may have special properties expressable in equational axioms that then constitute a variety.

4 Realization of a Holonic Multi-Agent System

We shall now descend from the previous level of abstraction and turn to finer grained issues that provide a basis for an actual implementation of a *holonic multi-agent system*.

4.1 Holonic Structures in Agent Societies

Let us first look at some general possibilities for modeling holonic structures and evaluate whether they are suitable for the design of a holonic system. The following notions differ in the degree of autonomy the sub-holons have and cover the spectrum from full sub-holon autonomy to a complete lack of autonomy.

A Holon as a federation of autonomous agents At one end of the spectrum is a model which assumes that the sub-holons are fully autonomous agents with their predefined architecture and the super-holon is just a new conceptual instantiation of the same generic agent architecture, whose slots are filled in dynamically by the sub-holons as the computation proceeds. Figure 3 displays this constellation.

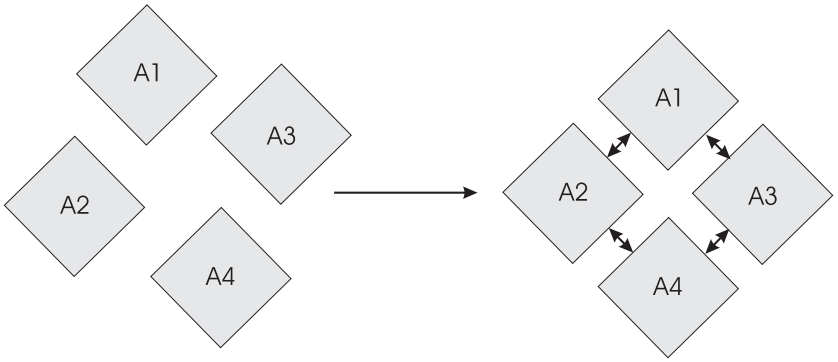

Figure 3: A holon as a federation of agents

In this case no agent has to give up its autonomy, and the super-holon is realized exclusively through cooperation among the sub-holons. The most transparent way of cooperation for this way is an *explicit coordination by commitment* via communication, i.e., agents negotiate over joint plans, task distribution or resource allocation. If commitments can not be established through communication, *implicit coordination* can be achieved in two ways: either, the holons are designed such that a goal directed common behavior emerges from the behavior of the sub-agents, or some sub-holons are able to represent goals and intentions of others and to reason about them; thus, they coordinate their actions without or at least with little communication.
The representation of a holon as a group of autonomous agents is in a sense just another way of looking at a traditional multi-agent system. The holon entity itself is not represented explicitly. In this case, holonic structures are only a design aid for structured agent-oriented programming.

Several agents merge into one The other extreme of the design spectrum would require to terminate the participating sub-agents and to create a new agent as the union of the sub-agents with capabilities that subsume the functionalities of the sub-agents (see Figure 4). In this case the merging agents completely give up their autonomy and their existence but they may be re-invoked into an existence of their own when the super-holon is terminated.

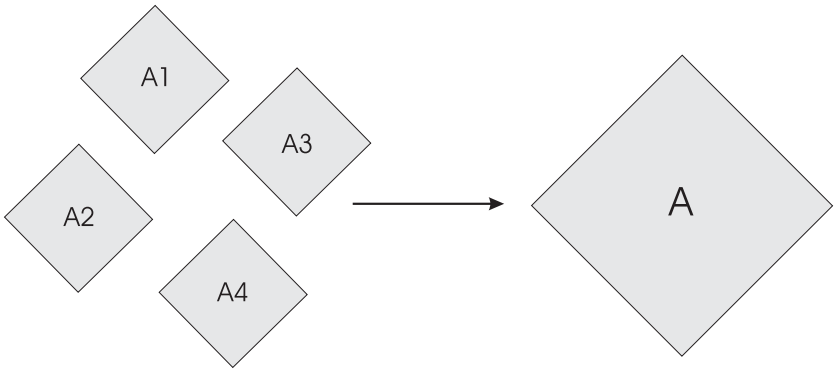

Figure 4: Several agents merge into one

The realization of this approach assumes procedures for splitting and merging holons that lead to the creation of a new agent. For agents of the same kind with an explicit representation of goals and beliefs (e.g., BDI agents) merging can be achieved by creating an agent with the union of the sub-agents' beliefs and goals provided consistency is guaranteed. Especially for a heterogeneous group of agents this can be intractable and in either case may not be very desirable. According to this model, agents cannot participate in more than one holon, unless they are copied.

A holon as a moderated group The two solutions above may be useful only under very specific circumstances. Instead, we shall propose a continuum, the extremes of which are the two above architectures. Consider a hybrid way of forming a holon, where agents give up only part of their autonomy to the super-holon which could be achieved by designation of one agent as a representative or *head* of the holon. This head represents the super-holon to the outside world, i.e. it provides the interface to the rest of the agent society. Its competence may range from purely administrative tasks to the authority to give directives to the other sub-holons. Furthermore, the head has the authority to allocate resources to the other agents in the holon, to plan and negotiate for the holon on the basis of its sub-agents' plans and goals, and even to remove some sub-holons or to incorporate new sub-holons. Figure 5 visualizes this approach.

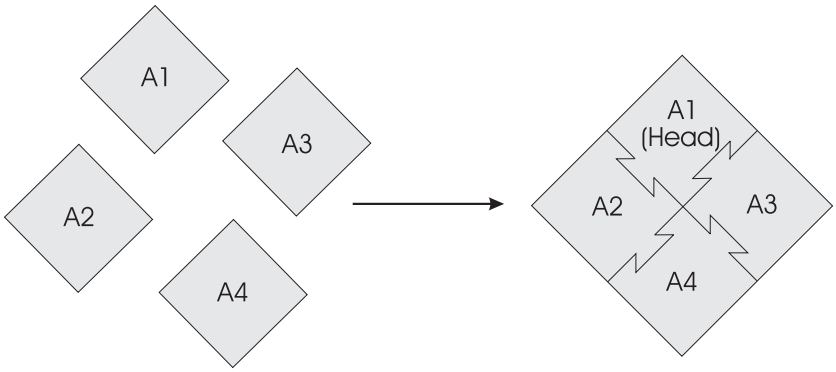

Figure 5: A holon as a moderated group

There are at least two methods to determine the head. Either, a new agent is created for the lifetime of the holon, or one of the members of the holon takes the role of the head and gains the additional functionality. In the second case either one member of the holon is a priori pre-destinated for the leadership or an election procedure is needed to promote one of the agents to leadership. Depending on the application domain, the competence of the representative may vary: the resulting structure can range from a loosely moderated group to a hierarchical structure. However, the members of the super-holon are always represented as agents, and, hence, we do not lose the capability to solve problems in a distributed fashion.

Conclusion Considering the strengths and limitations of these three approaches, we prefer the hybrid one: It allows for an explicit modeling of holons, a flexible formation of holonic groups, and a scalable degree of autonomy of the participating agents. The most challenging problem raising from this definition is the control of the individual and overall computation of the holonic multi-agent

system. We propose that control is established by resource bounded computation, i.e., computational resources are allocated within the holonic structures. This will be addressed in the following section.

4.2 Resource Allocation in Holonic Systems

We assume that every computation is relative to a given amount of resources. In the classic case this is just computational time and space (AM [Len76, LB84] was the first system to be based on this kind of control): when either time or space is consumed by the process, it is terminated and control passes back to the calling procedure.
We are working with a concurrent, constraint-based logical programming language called Oz [SSR95] and its environment MOZart [Smo98], which supports this style of computation. In this programming environment, the holonic system is given a certain amount of computational resources and the computation within that system is determined by the way these resources are distributed onto the sub-holons. Hence, the whole problem of distributed computation boils down to the problem of how the resources of the super-holon are distributed to its immediate sub-holons. Again, we shall first present several possible techniques for doing so and discuss their strengths and shortcomings. These techniques differ in their respective way *abstract resources* [GJ98] are distributed, where the spectrum of mechanisms ranges from totally decentral approaches over moderated ones to more centralistic mechanisms. But first let us introduce the concept of an abstract resource.

4.2.1 Informal Characterization of Abstract Resources

All mechanisms are based on the distribution of *abstract resources*: this concept is a generalization of the familiar notion in computer science, where resources are mainly computational time and memory space. We use the term abstract resource for *any environmental device or tool that enhances the behavior of an agent:* A resource (e.g., information, perception, capabilities) can enhance, or by its absence, constrain the agent's action. This notion includes not only classic resources such as time and space, but also, e.g., pieces of information, external tools or other agent's capabilities.[5] For instance, the construct of a *semaphore* is a classic control mechanism to handle resources: only one of the agents is able get a hold on this resource and is therefore allowed to compute or act.
From the point of view of the super-holon, the capabilities of its sub-holons can be treated as abstract resources as well. So, in a holonic society, abstract resources can also be, for instance, the capabilities of *sub-agents* that build a holon, the

[5]There is a collaborative research center (SFB-378) on resource-adaptive cognitive processes in Saarbrücken, funded by the German Basic Research Agency (DFG); see `http://www.coli.uni-sb.de/sfb378/index-en.html`.

number of specialists (i.e. the sub-sub-holons) for certain tasks, *different communication protocols*, etc.
Resources allocated to a super-holon are then redistributed at a finer granularity to its sub-holons, and this allocation can be viewed as a "guideline" of the computation for the lower ones. Of course abstract resources should be *typed* to ensure that only meaningful resources are passed on as parameter/value pairs.

4.2.2 Formal Definition of Resources

Let us extend the mathematical framework of Section 3.2 with a formal characterization of resources. To this end, we assign a value to each state of the environment, in order to express that some states are more desirable for a situated agent than others. As this utility value has to be computed by the individual agent, it depends on its local perspective and is therefore subjective, and, possibly, may include errors.
As we have shown above (Lemma 1), we can transform a multi-agent environment into a single agent environment. Hence, for the definition of utility functions it is sufficient to consider a single-agent environment $E = (E_1, \ldots, E_n)$ with single agent $\alpha = (S, P, A, \phi)$.

Definition 5 *A* utility function *is a mapping* $u : E \times S \to \mathbb{R}$ *where* $u(e)$ *denotes the value of the situation* e *for the agent* α*.*

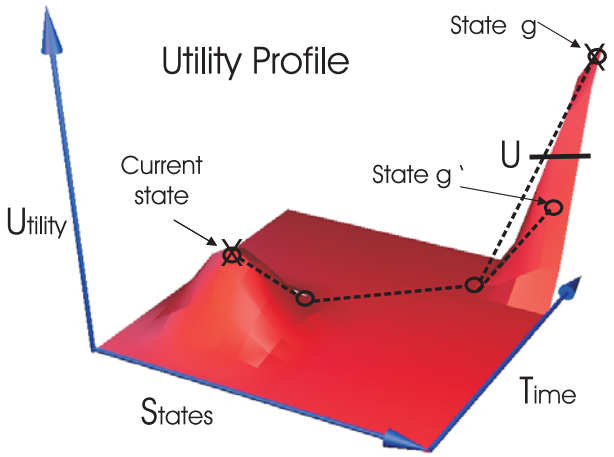

Figure 6: Utilities of different states

In Section 3.2, we have defined the environmental states as tuples of independent substates. Taking resources to be parts of the environment, we identify certain sub-sets of substates as resources.
Resources are that part of the environment that enables the agent to reach a state of the world that has a higher utility than the current one. In particular, we are

interested in sequences of actions that serve to achieve a state with higher utility, and we define those parts of the environment as resources that are essential for the success of these action sequences (see Figure 6). In the following definition we look at the positive case (the vital resource is available); the negative case (the resource is not available) is given in brackets.

Definition 6 *Let $e = (e_1, \ldots, e_i, \ldots, e_n, s) \in E \times S$ be an environmental state, and $u : E \times S \to I\!R$ be a utility function. A* resource *E_i is a part of the environmental state $E_1 \times \ldots \times E_n \times S$ with respect to U if*

- *there is a $U \in I\!R$ with $U > u(e)$ and*
- *there is sequence of actions $a_1, \ldots, a_n \in A^*$ that transforms (cannot transform) e into a state g with $u(g) \geq U$*
- *there is another $e'_i \in E_i$ such that $e' = (e_1, \ldots, e'_i, \ldots, e_n, s)$ cannot be transformed (can be transformed) into a state g' with $u(g') \geq u$ by the same sequence of actions.*

This definition corresponds to the informal characterization from above.

4.2.3 Resource Allocation Mechanisms

Resource allocation can be either controlled a priori by some central device or it can be controlled by the individual holon members through negotiation. We discuss various options to realize these resource allocation mechanisms and comment on their suitability for different settings. In particular, we distinguish between *cooperative* and *non-cooperative* settings: In a *cooperative setting*, the participants have no local utility valuation. They are eager to maximize the utility of the group, and, hence, a utility measure is needed that enables the agents to decide locally whether a trade is globally beneficial. In a *non-cooperative setting*, each agent tries to maximize its local utility. In this case, we must have the possibility for utility transfer via side-payments[6].

Market mechanisms Market-based mechanisms are used to distribute and redistribute tasks or resources among a holonic group of agents as long as they have roughly pre-defined schedules of tasks and resources. The general idea of these mechanisms (see [FMP96] for an introduction) is that each agent "advertises" a task or resource; the other agents bid for the resource or for the execution of the task which is then allocated such that the quality of the overall distribution increases.

[6] It might seem contradictory to consider self-interested members of a holon since we require a common overall goal. However, such a common goal does not prevent conflicting goals of minor priority.

We distinguish between coordinated and uncoordinated market mechanisms: in an uncoordinated market, agents negotiate and decide locally whether or not to agree on a deal. Sandholm [San96] proposed a trading mechanism to exchange tasks between agents; he showed that a globally optimal allocation is possible under certain circumstances. In a coordinated market a central instance, in our case the holon's head, moderates the trading process and provides the global utility function.
Simulated trading [BHM92] is a randomized algorithm that realizes such a market mechanism. The head collects the trading offers and evaluates them such that the global quality increases. The trading proceeds over several rounds, each of which consists of a number of decision cycles: in every cycle each agent submits one offer to sell or to buy a task. As in simulated annealing [KGV83], a relaxation value that decreases from round to round can be specified. If the algorithm terminates before a better solution is found, the best solution hitherto is returned, hence simulated trading is an anytime algorithm.

Game theoretic allocation mechanisms The head of a holon has to mediate between agents and to allocate the resources or tasks to a group of agents on the basis of reported valuations. Again, we can distinguish between the cooperative case (i.e., truthful behavior is guaranteed) and the non-cooperative case where the agents may try to increase their own benefit at the expense of others. In the latter case, for the sake of global performance, it might be useful to apply *truth revealing mechanisms*. Some sort of currency is needed that allows an explicit utility transfer, and the actual allocation mechanism could then be based on game theoretic techniques.
One of the classic protocols for cooperative settings is the *contract net protocol* [Smi80]. It assigns a task or resource to a single agent competing with a number of other possible contractors (the sub-holons, in our case). The manager (the holon's head) announces the resource or task to be allocated to the contractors which then submit a bid and state their cost of the bid. The manager grants the item to the bidder that stated the best offer and all other bids are rejected.
For a non-cooperative setting, *auction-based protocols* are better suited for the distribution of tasks and resources. Well-known protocols are the following: In the *sealed-bid-first-price* auction, all bidders submit a sealed bid and the bidder who offered to pay the highest price makes the deal and pays the price he actually bid for. In the *sealed-bid-second-price* auction (also called *Vickrey auction*) the bidder that submits the highest bid wins the competition but will only be charged the price the next bidder was willing to pay. The *English auction* is often applied in auction houses. Starting with the minimal price the auctioneer would accept, the bidders successively outbid each other until a single bidder is left. In the *Dutch auction* the auctioneer initially starts with a very high price which he lowers stepwise until one of the bidders accepts to buy the item at the current

price. The Vickrey auction is an *incentive compatible* mechanism: A bidder's dominant strategy is to reveal his real valuations to the auctioneer, i.e., it is well suited for a non-cooperative setting. The Vickrey auction is logically equivalent to the English auction, assuming a small step size in price increase.
All of these auction mechanisms are currently subject to game theoretic investigations of their properties, for some of these results and a survey with respect to multi-agent systems, see [FRV98].

Coordination based on heuristics Decentral mechanisms face sub-optimality, i.e., only local optima that may be less than the global optimum are achieved. Better solutions can be obtained if the decision is shifted to the holon's head, and the following approach extends the contract net protocol in that sense. Instead of announcing a task or a resource to the member agents and to let them evaluate it on the basis of their local information, the head requests the sub-agents' relevant local information. Collecting this information and using it for a central evaluation, the head obtains a more global picture for the resource/task allocation. It can then distribute the items on the basis of appropriate heuristics.
Organizing a holon can be viewed as an optimization problem by defining a *search space* and an *objective function* to be optimized. The objective function denotes the holon's performance (i.e., the global profile) while a multi-dimensional search space must describe the holon's resource distribution: Each type of resource reflects one dimension in the search space. If resource distribution can be undone, the representative can reallocate resources in order to increase performance. Hence, the *steepest ascent mechanism* can be applied for this kind of search space. Such an approach is presented in [Ger98].

Discussion These techniques can all be used for the resource and task assignment. They differ with respect to the degree of central control: Market mechanisms require no central control unit; the other approaches presuppose such a control unit whose competence, however, varies. In the non-cooperative setting, the head has only administrative competence and therefore mechanisms that enforce cooperative behavior have to be applied. In the cooperative case, decision power is split: the head decides on the basis of the local calculations of its sub-holons. Finally, in the central heuristic-based approach, the holon's head has the full resource allocation competence. Local information is provided by the sub-holon and is only used as a heuristic.
More decentral approaches are better suited to cope with complex allocation problems, as they can often be reduced to a set of problems with less complexity (divide-and-conquer). Nevertheless, the use of these methods may be less than optimal. Hence, the choice of the appropriate mechanism depends on the nature of the application and there is a trade-off between optimality and complexity.

5 Holonic Domains

A holonic multi-agent system may be too exaggerated for many traditional application domains, hence we examine, how domains can be characterized and delimited from those that are better suited for traditional multi-agent system. Having classified such *holonic domains* to some extent in the following paragraph, we discuss the suitability of the holonic scheme for a variety of applications in the second paragraph of this section.

5.1 Characteristics of Holonic Domains

Obviously it is not possible to give an absolute classification: the boundaries between domains that are suitable for holonic agents and those that are not, are blurred. So we will present a collection of criteria as a *guide* for the classification.

Operator abstraction Holonic systems are well suited for domains with actions of different granularity. Macro-level actions are carried out by the holon's head and decomposed onto the sub-holons. This could be realized in a traditional MAS also; however, the relationship between the individual agents and the group would have to be represented additionally; a holonic system provides all the relevant features a priori.

Hierarchical structure An application domain that exhibits a hierarchical structure is usually an excellent candidate for a holonic system, since hierarchies of sub-holons can be modeled canonically. The structure of the domain induces abstraction levels, which can be modeled naturally in a holonic system.

Decomposability One of the main pre-requisites for a traditional agent-based system is a decentralized or decomposable problem setting, where each agent is assigned to one of the sub-problems. Pro-activeness and autonomy of the agents are the main features.
However, often, problems are neither completely decomposable nor completely non-decomposable; in many *hybrid* cases, some aspects of the problem can be decomposed, while others cannot. Holonic agents are structured hierarchically, they can easily realize actions of different granularity, they are autonomous to a certain degree and they are pro-active; hence holonic agent systems can naturally deal with problems of that type.

Communication If the overall problem is decomposed into sub-problems that are not partitionings of the original one, but there is some overlap in the sense that logical interdependencies occur, communication among the problem solvers is needed. Sub-agents of a holon are communicative and hence, holonic agents

are useful in domains of this type. Furthermore, a domain often induces an unsymmetric communication behavior between problem solvers in the sense that each unit does not communicate to all other units equally often, i.e., patterns in the communication behavior can be observed. These patterns indicate possible structures for holonic agents: Holons provide facilities for efficient *intra-holonic* communication, supporting higher frequent communication inside the holon than among different holons (*inter-holonic*).

Social elements We have already distinguished between cooperative and non-cooperative settings. A cooperative setting does not constrain the use of holons in any way. However, in non-cooperative settings (e.g., virtual market places), things are different. If there is *no* cooperation among agents in the domain, the use of holonic agents is not very reasonable. If there are cooperative elements in the domain, holonic agents can be used to model the cooperative sub-domain.

Situatedness and real time requirement For many applications real-time behavior is a vital issue: The problem solver has to find a solution within limited computation time. As for some traditional agent architectures, we put the requirement of bounded rationality for all members of sub-holons to explicitly reason about time and other resources in order to find the best possible action within a given resource allocation.

Conclusion The most important requirements for a holonic agent are structure and cooperation: The domain should have a holonic structure, i.e, it should be recursively decomposable. This structure can then be be mapped canonically onto the holonic system. Furthermore, there must be sufficient cooperative elements between the distinguished problem solvers. One important difference to a traditional multi-agent domain is the possibility to also model centralistic aspects of a domain.
While these criteria are necessarily vague and general, we shall now turn to a selection of case studies and practical applications in order to apply these criteria.

5.2 Four Domains for a Holonic System

The following four application domains may serve for an evaluation of the postulated criteria for a holonic approach. For each of these examples we shall first discuss the domain and then present a holonic solution.

5.2.1 Transportation Scheduling as a Multi-Agent Domain

The domain Transportation tasks are planned and executed with a limited amount of transportation resources. A *transportation task* is a customer request to haul some goods from one place to another within specific time slots.

The haulage company has a limited number of *transportation units* like drivers, trucks, trailers or tractors at its disposal that must be combined into appropriate means of transportation, i.e. *vehicles*. The transportation units are not uniform but differ in many ways: The working time of a driver has legal constraints and also the type of cargo he is allowed to transport depends on his legal status. The trucks can be classified into trucks with or without loading space and in truck tractors. The type of the loading space constrains the type of cargo that can be transported, etc.
The *fleet scheduling problem* is a two-stage planning problem: (1) The transportation tasks fulfilling customer requirements including time constraints have to be assigned to the vehicles. (2) Vehicle tour plans for the assigned tasks have to be generated. Both sub-problems are known to be NP-hard and in fact even constrain each other leading to a further increase in complexity. Therefore, large order sets cannot be optimally scheduled within reasonable time. However, suboptimal and heuristic problem solving techniques for this problem are known from the field of Operations Research (see [GA88] for a survey), constraint programming (see [SW98]) and also, form multi-agent technology (see [FMP96]).
This domain meets the characteristics of a holonic domain as follows:

- Operator abstraction: The action in this domain is the execution of transportation tasks. Actions can however be defined at different levels of abstraction: the most general level specifies which transportation task a vehicle has to perform. These actions are then recursively decomposed into actions of lesser abstraction, such as loading, driving or vehicle maintenance, which again consist of sub-actions such as docking to a terminal, using traffic information systems, refueling, etc.

- Hierarchical structure: Transportation units must be combined to form vehicles.

- Decomposability: The fleet scheduling problem can be naturally divided into the subproblems of assigning a set of tasks to the vehicles and secondly of route planning for the vehicle fleet.

- Communication: Coordination among units that form a common vehicle requires a high amount of communication; cooperation among units of the same company that do not participate in the same vehicle need less; units belonging to different companies do not communicate in this scenario.

- Social elements: The setting is cooperative within a company and competitive between companies.

- Situatedness and real time requirement: Although in general there is plenty of time for tour planning (since planning takes much less time than tour execution), some situations require a fast and real-time answer, e.g., in case

of re-planning during execution time or when urgent orders are coming in and have to be scheduled immediately.

The solution In a traditional multi-agent based approach, the vehicles would be modeled as autonomous agents that compute local plans from which the global solution emerges (see [FMP96]). In a holonic modeling, such as the TELETRUCK system (see Section 7 for more details), the basic transportation units (trucks, trailers, drivers, chassis, and containers) are modeled as *component agents*. These agents merge into a holon that represents the vehicle for the transportation task. The vehicle holons are headed by a *PnEU (Planning 'n' Execution Unit)*, a special agent that is equipped with planning capabilities. The vehicle holons and the agents representing currently idle transportation units form a super-holon that represents the whole transportation company. The head of the company holon, called the *company agent* coordinates the interaction with the user and communicates with other companies that employ the TELETRUCK system. This modeling is in accord with the methodology of a *holon as a moderated group* as discussed in Section 4.1.
In the TELETRUCK system, there are four abstract resources: the *driving time* of the driver, the *loading space* of trucks and containers, the *chassis* that is supplied by components that can carry containers or swap bodies and finally, the *motor* resource. All these resources are necessary to actually execute a transportation task.
The TELETRUCK system allocates transportation tasks to the available transportation units such that the resource consumption is minimized. In this system, the *contract net protocol* and the *simulated trading procedure* (see Section 4.2.3) are used for resource and task allocation. All *agents* in the TELETRUCK realization follow our requirements for holonic agents of Section 3.

- Autonomy: Agents representing transportation units are autonomous in their decision to participate in a vehicle holon. Participating in the holon however restricts the autonomy of the sub-holons for this time span, since they have to execute the sub-tasks allocated to them.

- Common goal-directed behavior: The agents forming a vehicle holon cooperate in order to pursue the goal of executing a set of transportation tasks. Sometimes, even different vehicle holons cooperate for a task.

- Increased group capabilities: A vehicle holon is able to transport the cargo, which none of its components could do on its own.

- Belief: The agents have an explicit representation of the environment and the agent society.

- Bounded rationality: Because of the dynamics and the real-time requirements in this domain, an anytime algorithm is used for task allocation:

the run of the simulated trading procedure can be interrupted at any time and the current solution can be taken as a tour plan. Hence, the overall performance increases monotonically over time.

- Communication: In TELETRUCK, communication is structured in a hierarchical fashion. The company agent communicates with those agents that represent other companies, but also with the PnEUs, which in turn interact with the basic agents representing the components. Furthermore, in order to optimize the task and resource allocation, the PnEUs communicate with each other to exchange tasks. Communication among companies is exclusively performed by company agents.

The TELETRUCK system is one of the main application projects in this area at the DFKI, we shall present further details in Section 7 as a case study.

5.2.2 RoboCup

The domain The RoboCup Initiative defines and coordinates the "official Soccer world championships" for physical robots and software agents[7]. In the *simulation league*, each player is represented by a separate program that is connected via TCP/IP to a central simulation server. Every 100 milli-seconds, a player program can perform an action (dashing, turning, kicking, catching, communication with other players) by sending an appropriate string to the server which in turn computes the effect of that action. Such effects are transformed into the local perception of the agents and sent as percepts every 150 milli-seconds. Perception is more blurred the farther away entities are from the receiver. The server treats communication just like any other action; hence, long-distance communication is disturbed or not possible at all.
RoboCup is an holonic domain for the following reasons:

- Operator abstraction: The RoboCup'98 simulation engine treats all actions alike; they can be performed by any player program. However, there are also complex, strategic actions which involve several players (e.g., a double-pass). For the RoboCup'99 tournament it is planned to introduce a "coach" program for each team. This program will have a more global perspective of the scene and it can communicate with all agents of its team. Such a strategic communication action is more abstract than the other actions.

- Hierarchical structure: A priori, all players have equal status. However, it turns out to be of advantage it some players are designated to manage the offense block, the defense block, or the mid-field. At the next hierarchical level, the coach agent gives guidelines to these regional leaders.

[7]For details see `http://www.robocup.v.kinotrope.co.jp/02.html`

- Decomposability: The overall task to win a game can be decomposed into offense goals (to score many goals) and defense goals (to avoid scores of the opponent team).
- Communication: Players can communicate with only those players that are not too far away (in general, players of their region). The coach can communicate with all members of the team.
- Social elements: Within one team, the setting is cooperative, between teams it is of course competitive.
- Situatedness and real time requirement: This scenario is under strong real-time requirements, where a player program has to cope with rapidly changing percepts within 150 milli-seconds in order to determine the next action. Hence bounded rationality is an issue.

The solution The *CozmOz* [Jun98] team of the Universität des Saarlandes participated in RoboCup'98 and reached the quarter finals. Each player (and the coach) of this system is implemented as an InteRRaP agent [Mül96] and it is assigned to a region (offense, mid-field and defense). Some agents have additional control facilities for these regions (regional leaders). The coach has total control over all players. Now each player is modeled as a holon that joins into the super-holon of the *regional players* whose head is the *regional leader*. This super-holon in turn is a sub-holon of the holon *team*, whose head is the coach.
Each player controls a set of individual resources: *stamina*, for instance, simulates the "physical condition" of a player and this is controlled by the central simulation server of the RoboCup initiative. Every action of a player results in a decrease of stamina; a low stamina reduces the speed of an agent on the grid (the player gets more and more exhausted). Performing no action for a while increases the stamina (the player recovers). In the CosmOz system, stamina is modeled as an abstract resource the agent can reason about. Several threads inside an agent (representing e.g. the actions *move*, *kick* and *turn*) apply for the stamina resource (see Figure 7) and based on expected utilities of these actions, a fast and greedy mechanism allocates this resource.
Another abstract resource is *aim* and, again, several threads inside an agent (for example *aimSelf*, *aimPlayer* and *aimGoal*) apply for this resource. It is assigned on the basis of local utility measures and the winning thread performs the envisaged action, e.g., to shoot at the goal. The coach agent and the regional leaders control macro-level resources, such as the *roles of members in a block*, *tactic*, or *line-up*. For instance, if the team is only closely leading and the game is almost over, all player agents should be assigned to the defense.
Memory-based reasoning [SW86] is part of this architecture: Some situations an agent finds itself in are stored as *prototypical situations*. In case of a significant increase (*or* decrease) of the team performance or player performance (e.g., a

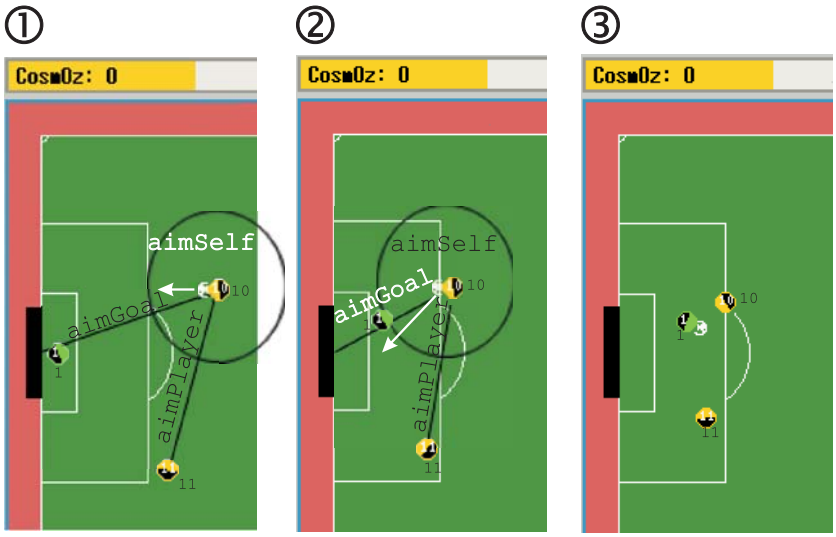

Figure 7: Threads applying for resources in the CosmOz team

goal is scored or a pass is intercepted), we store a 3-tuple of the *situation*, its *resource allocation* and effect on the *performance*. The current situation is matched against all prototypical situations; whenever a similar situation is found, the effect of the allocation chosen in that particular situation is taken as advice for the current situation: if the performance had increased, the resources are assigned in accord with the prototypical situation. A performance decrease in a prototypical situation is seen as an indication to assign the resource in the opposite way. (see [LJG99] for more details.)
Since this learning technique is a heuristic for resource allocation, it implements the *coordination approach based on heuristics* of Section 4.2.3. Let us now look at our holon requirements:

- Autonomy: The coach and all players are independent programs that communicate over TCP/IP, hence agent autonomy is predefined. However, as players are commanded by the coach and the regional leaders, their autonomy is restricted in the sense of our holon definition.

- Common goal-directed behavior: Obviously, the team members have a goal they all strive for: to win a game (and to become world champion eventually).

- Increased group capabilities: At the highest level, the team level, the execution of strategies (e.g., playing in an offensive or defensive style) is realized by the whole team. Cooperative actions such as double-pass are defined at the next lower level, but still no single agent could execute them alone.

- Belief: Every player agent has to maintain its own world representation (its own position and stamina, the relative position of the ball, etc.). At the group level, the coach agent represents the state of the whole team and of the opponent team (score, tactic, time to play, average stamina, etc.).

- Bounded rationality: Our resource-adaptive agents display bounded rationality in the sense that the abstract resources define limits for the individual computation.

- Communication: In the RoboCup setting, communication is restricted to agents within a certain geographical distance. However, communication to the head of the holon, i.e. the coach, is always possible. Communication to agents outside of the holon (the players of the other team) is not provided.

In summary, all requirements for a holonic multi-agent approach are fulfilled and this domain is surely holonic in nature. In particular dynamic strategies to form a group (such as double-pass, offside trap) can be realized easily by a dynamic configuration and reconfiguration of holons.

5.2.3 Flexible Manufacturing Systems

The domain Job-shop scheduling of work in a production plant must be optimized in a manufacturing process. A vital issue is the problem of *dynamically scheduling* a production plan, as it cannot be guaranteed in general that a schedule will be fulfilled: workstations may fail, supply parts may be out of stock, workers might not show up for work or may be injured during their work time etc. Let us see if this problem domain is holonic:

- Operator abstraction: Obviously, there are plenty of actions at different levels of abstraction: At the very bottom level, actions are e.g., screwing and welding, higher level actions are the integration of modules to the chassis or at the highest level of abstraction, assembling a product.

- Hierarchical structure: There are basically two types of entities: workstations (with human workers and automated cells) and chassis (consisting of smaller modules). Each of the two imposes a a rather flat hierarchy.

- Decomposability: The overall problem of controlling a manufacturing plant can be nicely decomposed into subproblems: If a workstation fails, only the schedules of the affected chassis have to be modified.

- Communication: The domain imposes no restrictions on communication among entities. Communication between workstation representatives and chassis representatives will be necessary when the chassis from a broken workstation must be detoured to one in function.

- Social elements: Clearly, this domain is strictly cooperative.
- Situatedness and real time requirement: A centralized re-planning of the whole schedule is often not possible, however, local re-scheduling may be feasible since this is a question of seconds while the system runs in terms of minutes and hours. Hence, bounded rationality is not really an issue if the replanning is decentralized.

In summary, most criteria are fulfilled in the setting, in particular the two most important ones, namely hierarchical structure and cooperation. However, depending on the degree of detail of the model, it may be reasonable not to model all entities in that domain by (holonic) agents as we shall discuss next.

The solution The *IFMS (Intelligent Flexible Manufacturing System)* [BFG99] has been developed in cooperation with experts from a major German car manufacturing company. The key idea of IFMS is to represent every workstation (and a buffer, i.e., a station where chassis are just temporarily stored) and every work piece by some agent that plans and monitors the local schedule of its station or piece. If for some reason the current schedule cannot be executed, agents re-plan the schedule in a decentralized manner (See Figure 8).

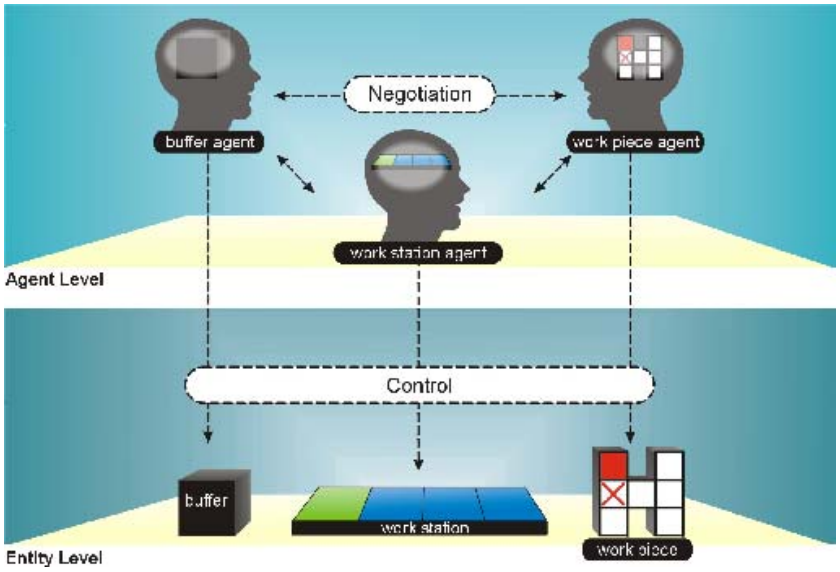

Figure 8: Agents negotiating over holon membership in IFMS

We consider a chassis as a holon that exists as long as the processing of that chassis lasts. After an initial schedule has been computed (prior to the actual assembly), a chassis is assigned to the set of workstations, which are involved in the manufacturing process. The holon consists of the chassis agent as its *head*,

which controls the assembly of the product, while the workstation agents are the *sub-holons*. A more detailed representation (e.g., the agentification of the automated parts or the human workers) is not necessary in this case, leading to a rather flat hierarchy. Note, that a workstation agent can be a member in quite a number of holons at a time.
The production tasks are viewed as abstract resources to be allocated to the holon members. Furthermore, *station functionalities* and *idle time slots* of workstations are also resources that have to be managed. If during the assembly process one of the workstations fails, its representative agent leaves the holon (since it can no longer provide the required resource), and the head has to find a substitute by announcing its need to all workstation agents. These agents evaluate their extra effort on a local basis and send a proposal to the head which selects the best offer and invites that workstation agent to join the holon in accord with the contract net protocol principle.
Is this model a holonic solution?

- Autonomy: All agents are autonomous, in fact, there is no real restriction of autonomy of any holon member.
- Common goal-directed behavior: All agents have the common goal to finish the assembly of the chassis the head is assigned to.
- Increased group capabilities: In general, the construction of a work piece cannot be performed by a single workstation as every workstation assembles only those parts it is specialized for.
- Belief: The internal knowledge of the member agents is left untouched by the head. The reasoning of the head is at a higher level of abstraction, namely on facilities of a set of workstations.
- Bounded rationality: This domain does not really require bounded rationality for the agents involved.
- Communication: Member and candidate member agents communicate only with the head, not with other workstation agents.

To summarize, as most of the requirements are fulfilled, we could model this flexible manufacturing system as a holonic agent system in a holonic domain. However, since such a system could also be realized with a regular agent approach, the holonic principle works more as a structuring aid than a real necessity.

5.2.4 The Coordination of Business Processes in a Virtual Enterprise

The domain A *virtual enterprise* (VE) [AFHS95] is a temporary federation of otherwise legally independent companies. Usually companies form a VE when

they identify a short-term market opportunity that none of the partners could exploit alone. The partners of a VE contribute their core competences to the common business processes, hence the VE is usually able to provide services or products of high qualities within a minimal respond time.
A virtual enterprise is not institutionalized, it has no employees or offices of its own. Nevertheless, the partner companies of a VE act as a single corporation when seen externally.
The coordination of the business processes within a VE, especially the coordination of manufacturing processes or the supply chain management, is a more challenging task than the management of a classical firm. Since there is no hierarchy in a VE, competence to divide what to do is often unresolved among the partners with clashing economic interests. Furthermore, by their definition and purpose, virtual enterprises have to react instantly to dynamic changes in the market.
A software solution represents the knowledge of the VE at several levels of abstraction, it has to plan and supervise business processes among several companies, and last, but not least, the precise allocation of tasks and resources has to be administered.
The above transportation scheduling and flexible manufacturing domains can be seen as special instances of business process management in general virtual enterprises. This domain is naturally holonic:

- Operator abstraction: Any kind of business process can be modeled as an action, ranging from very elementary tasks to complex procedures.

- Hierarchical structure: VEs have no institutional hierarchy. However, the companies a VE consists of usually do. Since the structure of partner companies can be mapped onto agent systems, the overall structure of a VE can be naturally modeled using a hierarchy in which the agents that represent the higher levels of the VE have only administrative rights.

- Decomposability: The tasks of the VE are the planning, distribution, and execution of business processes. These processes can be decomposed in elementary actions.

- Communication: There is a high degree of interaction required in order to coordinate the work of organizational units in companies and virtual enterprises.

- Social elements: The setting has cooperative elements as well as competitive ones. In principle, the partners within a VE aim at a common goal which is the reason for their forming a VE. Nevertheless, the allocation of tasks and the distribution of profit is competitive. Even inside a single company, there are competitive situations when sub-units have to compete for limited resources.

- Situatedness and real time requirement: As mentioned above, VEs have to react fast.

A VE can be modeled holonically as follows: the organizational structure of the VE is modeled as a holon, whose sub-holons are the individual companies that in turn are decomposable into sub-sub-holons that represent the different departments or subsidiaries. The business processes themselves are decomposable, and hence, can be modeled as holons as well. Companies without a hierarchy can be represented as holons in which the head has only a rather limited, moderating competence.
The exhaustive modeling of all processes in a VE by a holonic multi-agent system is a large and visionary task. But this task becomes more tractable when it can be divided into separate processes/holons that are then linked together. For example a multi-agent manufacturing system and a system for supply chain management implemented in the same holonic framework could be linked at a higher level into a system that is able to coordinate the supply with the manufacturing process.

6 Towards Holon Oriented Programming

In this section we propose a generic framework for the implementation of holonic agent systems, where the holonic structure of the agent society is explicitly represented. This framework provides the architectural structure for the TeleTruck system which is presented in detail in the next section.
We presuppose in the following any concurrent, object-oriented programming platform; our generic holon framework is implemented in the programming language Oz [SSR95]. An agent is realized by at least one object and one computational thread. Our implementation is based on Müller's three-layered InteRRaP architecture [Mül96] where each layer is modeled by an object and the processes operate concurrently within the layers.

6.1 The Agent Architecture

InteRRaP consists of three concurrently executing layers. The behavior-based layer (BBL) supplies the basic behaviors of the agent and handles immediate reactions to simple stimuli from the environment within a short respond time. The local planning layer (LPL) is responsible for the deliberative decision making of the agent; it controls the local planning and the configuration of the BBL. The topmost layer is the cooperative planning layer (CPL) which organizes the interaction with other agents. All three layers have access to the agent's knowledge base (KB) that represents the agent's world model. The world interface (WIF) translates an agent's abstract actions into physical actions and transforms low-level perceptions into abstract ones.

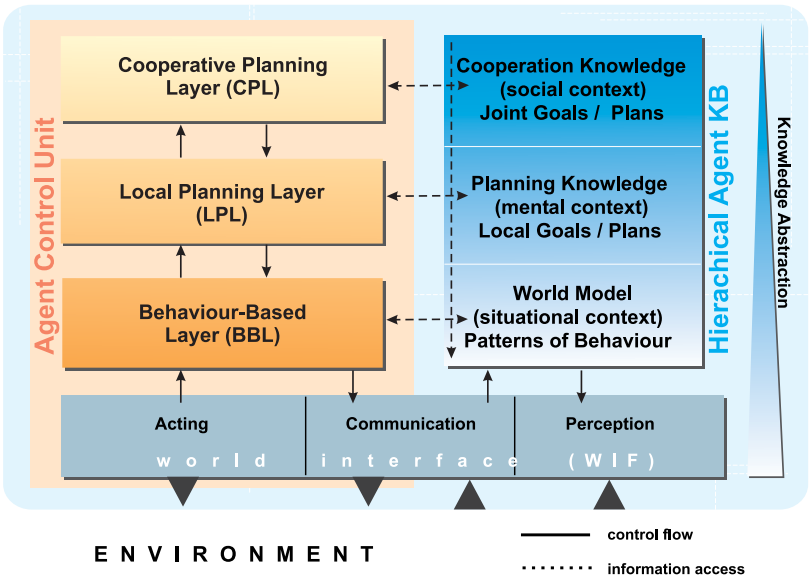

Figure 9: The InteRRaP architecture

The InteRRaP architecture is the basis for every member agent of a holon in our framework: every holon and sub-holon consists of a private CPL, LPL, BBL, WIF, and KB. The precise description of the generic holon implementation is out of the scope of this paper, we refer to the technical report [GV99]. The holonic structure of the agent society is represented as a directed graph of pointers that is maintained in a distributed fashion by the CPLs of the member agents.

6.2 Extensions of the Cooperative Planning Layer

The composition and configuration of holonic structures within an agent society must clearly be performed in the CPL which provides the functionality for communication, negotiation and the administration of holonic structures. Communication among agents that are not part of the same holon is organized via communication protocols. The CPL can concurrently manage several protocol instantiations that are represented as objects. In these objects the state of a negotiation is stored. All communicative acts are speech acts according to the KQML (knowledge query and manipulation language) [FF94].
Each holon is represented by one *holon object* (see Figure 10) that is maintained by the CPL of the holon's head. The functionality of the holon object is to store the structure of the holon in *reference-lists* with references to the holon's parts and head. Furthermore, the head administers a list of *authorities* which maintains the access rights to the methods of the holon object, and, optionally the rights to use communication channels. The holon object also stores *incompatibility lists* that maintain information about the holon's parts and other agents, for example,

the ability of two agents to merge or not to merge to one and the same holon. The holon object also provides a number of methods for the incorporation, removal or modification of sub-holons. These methods are exclusively accessible by the holon's head.

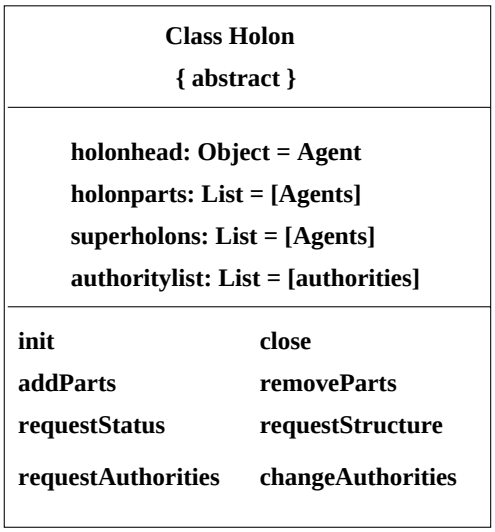

Figure 10: The holon object

The sub-holons may request information about the holon's structure, and about their own status and authorities from the holon object. Every agent maintains the references to the holon objects of the holons in which it participates and has methods for its incorporation into a holon and its removal from a holon. The incorporation has to be acknowledged by the agent that is to be integrated while the removal needs not to be acknowledged. This basic functionality may have to be extended for some applications.
For an efficient communication among the sub-holons, the designer can introduce shared logic variables between the sub-holons. Conceptually, this amounts to overlapping internal states of the agents, which is a violation of the autonomy requirement. This sacrifice may be justified if it is conceptually consistent and leads to an increase in performance. The use of a shared memory can be seen as a partial and reversible merge of the agents involved. If the overlapping is not total — which is usually the case — agents can participate in more than one holon.

7 A Case Study: TeleTruck

We have applied the holonic agent methodology in a multi-agent fleet scheduling system, called TeleTruck that has been developed at DFKI [BFV98a, BFV98b,

BFV99, GRV99]. The TeleTruck system models the business processes of a transportation company, in particular the on-line allocation of transportation requests. The company has a fixed number of transportation units like drivers, trucks, or trailers which have to be scheduled continuously at minimal costs.[8]

7.1 The Technical Environment

The primary service of a haulage company is the execution of transportation tasks for its customers. Consequently, an important business process of the company is the distribution of transportation orders onto its vehicle fleet and the generation of tour plans for the trucks, taking into account several constraints, like time constraints or incompatibilities between tasks or between tasks and vehicles. This problem is of exponential computational complexity, and, hence, cannot be solved optimally for large sets of transportation tasks. Nevertheless, in the highly competitive transportation industry there is an increasing demand for rationalization and optimization.
Figure 11 illustrates the technology that is already in use in many shipping companies and that provides an appropriate basis for tour planning software. The trucks are equipped with on-board computers which are linked on-line (via a mobile phone modem) or off-line (with some kind of docking facility) to the company's computer. The on-board computer supports the truck driver with the customer formalities on-site. It manages the tour plans and it protocols the execution of the plans. If the on-board computer is linked to the company on-line, the TeleTruck system informs the driver about last-minute modifications of the tour plan even during the plan execution. Furthermore, the truck can be equipped with a global positioning system (GPS), that allows to locate the vehicle precisely. The position of the vehicle is displayed on an active road map on the screen of the company computer. Besides the tracking of the vehicles, and, hence, the cargo, a route planning and optimization system can be applied that has to be integrated into the standard software of the company in order to achieve an optimal work-flow.

7.2 The System Architecture

Figure 12 shows the system architecture of TeleTruck. The central module is a holonic multi-agent system that manages the planning and optimization of the vehicle configuration and the tour plans. The multi-agent system as well as the user access a common SQL-database. The user, usually the dispatch officer of the company introduces tasks to the system, he can generate tour plans by

[8]The TeleTruck system, as well as the InteRRaP architecture is the outcome of several large research and development projects with overall costs that exceed by far the amount of two million Euro; in this proposal we shall just concentrate on the conceptual developments as far as they relate to the holonic paradigm.

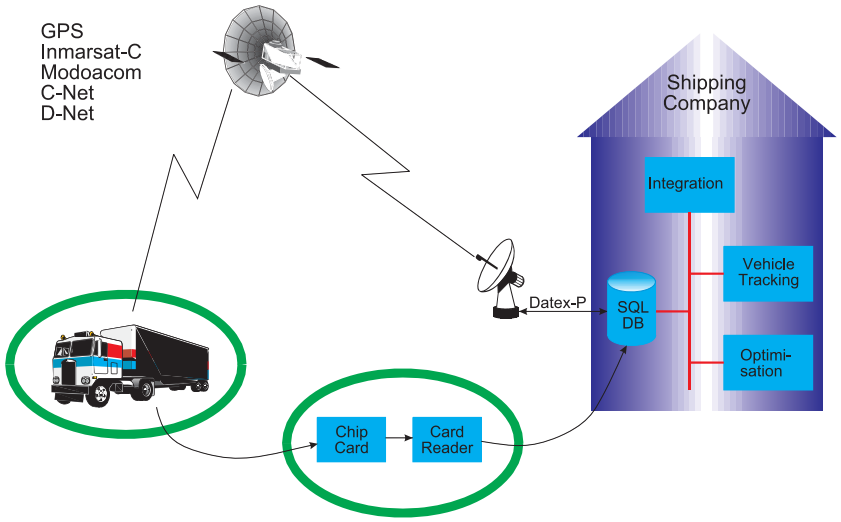

Figure 11: The technical environment of TeleTruck

hand or by delegating the planning to the multi-agent system. The officer can modify the plans generated by the system and can request an optimization of the solution. Furthermore, he can impose additional constraints, for example to book an order to a truck or a time slot. The on-board computer of the truck receives the GPS information and updates the position of the truck in the database. Electronic maps and routing software are added to supply the tour planning system with geographical data, which are commercially available today. We also plan to integrate on-line traffic information to react timely to traffic jams etc. The electronic maps have been supplied by our partners. The user interface and the agent system have been developed at DFKI.

7.3 Structure of the Holonic Society in TeleTruck

For each transportation unit of the forwarding company there is an agent that administrates its resources. These holonic agents have plans, goals, and communication facilities in order to provide their resources for the transportation plans. The agents can merge with a *Plan'n'Execute Unit* (PnEU) and form a holon that represents a complete vehicle. For example, a vehicle holon may consist of a PnEU, a driver, a truck, a trailer, and two containers, each being modeled as a sub-holon that is merged into the super-holon as shown in Figure 13.
The PnEUs are special sub-holons which coordinate the formation of the super-holon and plan the vehicles' routes, loading stops, and driving times. The PnEU is the head of the vehicle holon, represents it to the outside world, and is authorized to reconfigure it. A PnEU is equipped with planning, coordination, and communication abilities, but does not have physical resources. Furthermore, in

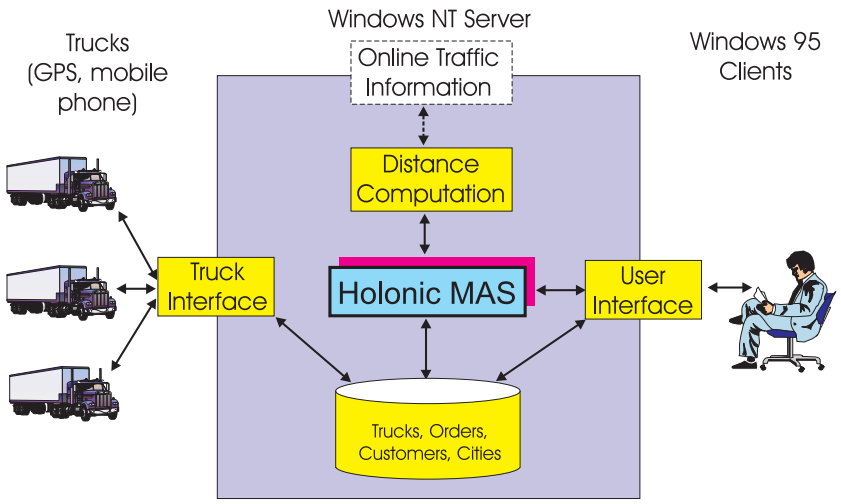

Figure 12: The TeleTruck architecture

the agent society, there is always at least one idle PnEU with an empty plan that represents the currently idle transportation units and coordinates the formation of a new holon from idle components.
The vehicle holons are in turn sub-holons of the super-holon that represents the entire transportation company. This holon is headed by a *company agent*, which announces and distributes the incoming orders, accepts the tenders, controls global optimization, coordinates the execution, and handles all communication with the user, i.e. the dispatch officer. It also coordinates the internal cooperation and interaction between the PnEUs.

7.4 Dynamic Holon Formation and Reconfiguration

For the formation and coordination of the holonic agent society we have chosen the *extended contract net protocol* (ECNP) [FMP96] and the simulated trading market mechanism. The ENCP splits a task into subtasks and distributes them to more than one contractor. We use the ECNP to generate an initial holon configuration and to allocate tasks to these holons. The tasks that are introduced to the system are passed to the company agent which announces them to the PnEUs. The PnEUs heading an existing vehicle holon check whether the resources of their components are sufficient for the execution of the task. If so, they compute the cost of the execution and submit an appropriate bid to the company. The PnEU that represents the idle transportation units and PnEUs without enough resources try to collect components that supply the missing resources.
In Figure 13, a company agent announces a new transportation task to two vehicle holons and the idle PnEU. The PnEUs heading a holon request the necessary

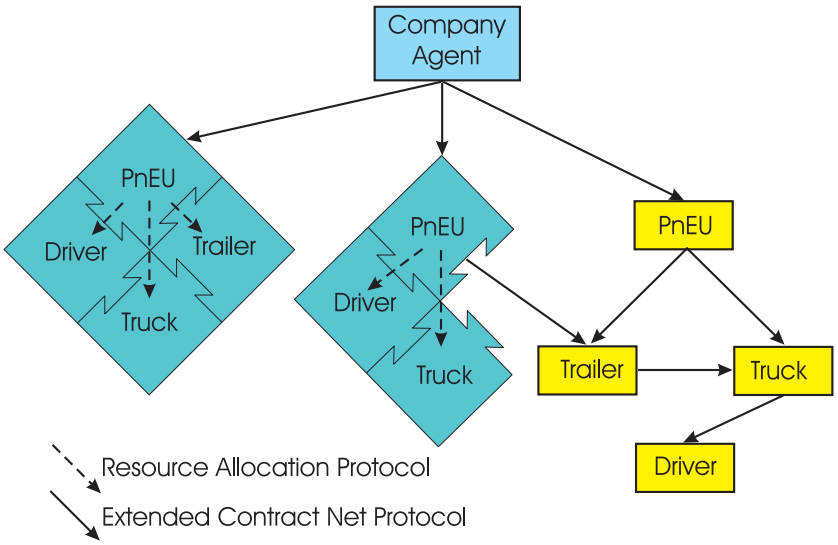

Figure 13: Holonic agents in TeleTruck

resources from their sub-holons and if possible, calculate the cost for the execution. The already completed holon on the left hand side cannot incorporate further sub-holons. The second vehicle holon in the middle of the figure can integrate a trailer. Hence, if its own resources are not sufficient, the head tries to collect the missing resources by performing an ECNP with the idle trailers that supply such resources. The idle PnEU first performs an ECNP with those idle components that offer loading space; in the example a truck and a trailer.
The trailer supplies loading space and chassis, therefore, it needs a motor and announces the task to the truck. The truck which received two different announcements for the same task—one by the trailer and one by the PnEU directly—can bid in both protocols since it can be sure that only one of the protocols will be successful. Therefore, the truck agent looks for a driver, computes the cost for the two different announcements, and gives a bid both to the PnEU and to the trailer.
Obviously, the cost for executing the task with a vehicle consisting only of a driver and a truck is less than the cost of executing the same task with the same truck and driver and, in addition, a trailer. Therefore, the idle PnEU will pass the bid to the company agent. If the task is granted to the idle PnEU, the PnEU merges with the components to a vehicle holon and a new PnEU will be created for further bidding cycles. Whenever the plan of a holon is finalized and executed, the components separate and the PnEU terminates.

8 Conclusion and Outlook

The main advantage of the holonic approach is the chance to recursively map an application domain directly and naturally onto a multi-agent system where the agents are again composed of agents.
Currently, we work on applications for intermodal transportation, for telematics, and for manufacturing systems. The further development of a generic holonic agent toolkit that supports *holon-oriented programming* as an extension of agent-oriented programming is another research goal.

Acknowledgments

The authors gratefully acknowledge fruitful discussions with Hans-Jürgen Bürckert and Klaus Fischer. Thanks to CPN GmbH, Konz GmbH & Co. KG, PTV GmbH, and Simac Systems bv for their support, their problem specifications and their supply of electronic maps and telecommunication systems. TeleTruck has been funded by the European Commission and by the local government of the Saarland. This work was also supported by the Siemens AG and the German Federal Ministry of Education, Science, Research, and Technology under grant number FKZ ITW-95004.

References

[AFHS95] O. Arnold, W. Faisst, M. Härtling, and P. Sieber. *Handbuch der modernen Datenverarbeitung*, volume 185 of *Theorie und Praxis der Wirtschaftsinformatik*, chapter Virtuelle Unternehmen als Unternehmenstyp der Zukunft? Heidelberg: Hüthig-Verlag, 1995.

[BFG99] T. Bohnenberger, K. Fischer, and C. Gerber. An agent-based approach for production scheduling and plant topology optimization in manufacturing. In *Proceedings of the Fourth International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents (PAAM)*, 1999.

[BFV98a] H.-J. Bürckert, K. Fischer, and G. Vierke. Teletruck: A holonic fleet management system. In *Proceedings of the 14th European Meeting on Cybernetics and Systems Research*, 1998.

[BFV98b] H.-J. Bürckert, K. Fischer, and G. Vierke. Transportation Scheduling with Holonic MAS – The TeleTruck Approach. In *Proceedings of the Third International Conference on Practical Applications of Intelligent Agents and Multiagents (PAAM'98)*, 1998.

[BFV99] H.-J. Bürckert, K. Fischer, and G. Vierke. Holonic fleet scheduling with teletruck. In *Proceedings of the Second International Conference on Computing Anticipatory Systems (CASYS'98)*, 1999.

[BHM92] A. Bachem, W. Hochstättler, and M. Malich. Simulated Trading: A New Approach For Solving Vehicle Routing Problems. Technical Report 92.125, Mathematisches Institut der Universität zu Köln, Dezember 1992.

[Bra87] M. E. Bratman. *Intention, Plans, and Practical Reason.* Harvard University Press, Cambridge, Mass., 1987.

[Cas95] C. Castelfranchi. Guarantees for autonomy in cognitive agent architecure. In M. Wooldridge and N.R. Jennings, editors, *Intelligent Agents: Theories, Architectures and Languages*, volume LNAI Volume 890, pages 45–70. Springer-Verlag, Heidelberg, Germany, 1995.

[CL90] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. In *Artificial Intelligence*, volume 42, pages 213–261. 1990.

[Dee94] S. M. Deen. A cooperation framework for holonic interactions in manufacturing. DAKE Centre, University of Keele, 1994.

[FF94] T. Finin and R. Fritzson. KQML — a language and protocol for knowledge and information exchange. In *Proceedings of the 13th Intl. Distributed Artificial Intelligence Workshop*, pages 127–136, Seattle, WA, USA, 1994.

[FMP96] K. Fischer, J. P. Müller, and M. Pischel. Cooperative transportation scheduling: an application domain for DAI. *Journal of Applied Artificial Intelligence. Special issue on Intelligent Agents*, 10(1), 1996.

[FRV98] K. Fischer, C. Ruß, and G. Vierke. Decision Theory and Coordination in Multiagent Systems. Research Report RR-98-02, DFKI, 1998.

[GA88] B. L. Golden and A. A. Assad, editors. *Vehicle Routing: Methods and Studies.* Elsevier Science Publishers, 1988.

[Ger98] C. Gerber. Bottleneck Analysis for Self-Adapting Multi-Agent Societies. In *Proceedings of the IEEE Joint Conference on the Science and Technology of Intelligent Systems*, 1998.

[GJ98] C. Gerber and C. Jung. Resource Management for Boundedly Optimal Agent Societies. In *Proceedings of the ECAI Workshop on Monitoring and Control of Real-Time Intelligent Systems*, 1998.

[GRV99] C. Gerber, C. Ruß, and G. Vierke. On the Suitability of Market-Based Mechanisms for Telematics Applications. In *Proceedings of the International Conference on Autonomous Agents (Agents'99)*, 1999.

[GV99] C. Gerber and G. Vierke. A gereric framework for the implementation of holonic agents. Technical report, DFKI, 1999. to appear.

[Jun98] C. G. Jung. Experimenting with layered, resource-adapting agents in the robocup simulation. In *Proc. of the ROBOCUP'98 Workshop*, 1998.

[KGV83] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimalization by simulated annealing. *Science*, 220:671, 1983.

[Koe67] A. Koestler. *The Ghost in the Machine.* Hutchinson, 1967.

[Lan98] D. Lange. Mobile agents: Environments, technologies, and applications. In *Proceedings of the Third International Conference on Practical Applications of Intelligent Agents and Multiagents (PAAM'98)*, 1998.

[LB84] D. Lenat and J. Brown. Why AM and EURISKO appear to work. *Artificial Intelligence*, 23(3):185–250, 1984.

[Len76] D. Lenat. AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search. Memo STAN-CS-76-570, Stanfort University, 1976.

[LJG99] J. Lind, C. Jung, and C. Gerber. Adaptivity and Learning in Intelligent Real-time Systems. In *Proceedings of the International Conference on Autonomous Agents (Agents'99)*, 1999.

[Min86] M. Minsky. *The Society of Mind.* Simon and Schuster (Touchstone), 1986.

[MSR99] J.P. Müller, M.P. Singh, and A.S. Rao, editors. *Intelligent Agents V. Agents Theories, Architectures, and Languages*, volume 1555 of *Lecture notes in artificial intelligence.* Springer, 1999.

[Mül96] J. P. Müller. *The design of intelligent agents: a layered approach*, volume 1177 of *Lecture notes in artificial intelligence.* Springer, Berlin, 1996.

[MWJ97] J.P. Müller, M.J. Wooldridge, and N.R. Jennings, editors. *Intelligent Agents III. Agents Theories, Architectures, and Languages*, volume 1193 of *Lecture notes in artificial intelligence.* Springer, 1997.

[Par69] T. Parsons. *Politics and Social Structures.* Free Press, New York, 1969.

[RG91] A. S. Rao and M. P. Georgeff. Modeling Agents Within a BDI-Architecture. In R. Fikes and E. Sandewall, editors, *Proc. of the 2rd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pages 473–484, Cambridge, Mass., April 1991. Morgan Kaufmann.

[RN95] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, 1995.

[RW91] S. J. Russell and E. H. Wefald. *Do the right thing: studies in limited rationality.* MIT Press, 1991.

[San96] T. Sandholm. *Negotiation among Self-Interested Computationally Limited Agents.* PhD thesis, University of Massachusetts at Amherst, Department of Computer Science, 1996.

[Sho91] Y. Shoham. Agent-oriented programming. In *Proc. of the 11th International Workshop on DAI*, pages 345–353, 1991.

[Smi80] R.G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. In *IEEE Transaction on Computers*, number 12 in C-29, pages 1104–1113, 1980.

[Smo98] Gert Smolka. Concurrent constraint programming based on functional programming. In Chris Hankin, editor, *Programming Languages and Systems*, Lecture Notes in Computer Science, vol. 1381, pages 1–11, Lisbon, Portugal, 1998. Springer-Verlag.

[SRW98] M.P. Singh, A.S. Rao, and M.J. Wooldridge, editors. *Intelligent Agents IV. Agents Theories, Architectures, and Languages*, volume 1365 of *Lecture notes in artificial intelligence*. Springer, 1998.

[SSR95] G. Smolka, C. Schulte, and P. Van Roy. PERDIO: Persistent and Distributed Programming in Oz. Technical report, German Research Center for Artificial Intelligence, Saarbrücken, March 1995.

[SW86] Stanfill and Waltz. Towards memory-based reasoning. *Communications of the ACM*, 29(12), 1986.

[SW98] Klaus Schild and Jörg Würtz. Off-line scheduling of a real-time system. In K. M. George, editor, *Proceedings of the 1998 ACM Symposium on Applied Computing, SAC98*, pages 29–38, Atlanta, Georgia, 1998. ACM Press.

[WJ95] M. J. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

Holonic Multi-Agent Systems

Christian Gerber, Jörg Siekmann, Gero Vierke

RR-99-03
Research Repor