Interaction between Agents in Routine, Familiar and Unfamiliar Situations *

B. Chaib-draa
Département d'informatique, Faculté des Sciences
Université Laval, Sainte-Foy, QC, G1K 7P4, Canada
e-mail: chaib@ift.ulaval.ca

Abstract

A framework for designing a multiagent system (MAS) in which agents are capable of coordinating their activities in routine, familiar, and unfamiliar situations is proposed. This framework is based on the skills, rules and knowledge (S-R-K) taxonomy of Rasmussen. Thus, the proposed framework should allow agents to prefer the lower skill-based and rule-based levels rather than the higher knowledge-based level because it is generally easier to obtain and maintain coordination between agents in routine and familiar situations than in unfamiliar situations. The framework should also support each of the three levels because complex tasks combined with complex interactions require all levels. To permit agents to rely on low levels, a suggestions is developed: agents are provided with social laws so as to guarantee coordination between agents and minimize the need for calling a central coordinator or for engaging in negotiation which requires intense communication. Finally, implementation and experiments demonstrated, on some scenarios of urban traffic, the applicability of major concepts developed in this article.

Keywords: Agent architecture, coordination between agents, multi-agent systems.

1 Introduction

With the steady progress of research in information technology over the past decade, it is now clear that we are nearing the boundaries of current engineering approaches and that many classes of complex problems cannot be solved in isolation. Research advances in distributed artificial intelligence (DAI), however, have opened up many new avenues for solving such problems [16, 35]. Generally, the DAI field aims to construct intelligent systems which interact one with another [7]. One of the major distinctions in this field is between research in distributed problem solving (DPS) [10, 11], and multiagent systems (MAS) [18, 28, 32].

The goal of DPS is to create a net of coarse-grained cooperating agents that act together to solve a single task, such as monitoring a network of sensors [11]. In DPS generally, the problem is divided into tasks, and intelligent solvers are designed to solve these tasks. All strategies of cooperation or coordination are incorporated as an integral part in the design of the system.

*This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by the Fonds pour la Formation des Chercheurs et l'aide à la Recherche (FCAR) du Québec.

In a multiagent environment, the agents are autonomous, potentially preexisting and typically heterogeneous. Research here is concerned with coordinating intelligent behaviors among a collection of autonomous agents, that is, how these agents can coordinate their knowledge, goals, skills, and plans jointly to take action and to solve problems [5, 7]. In this type of environment, the agents may be working toward a single global goal, or toward separate individual goals that interact. Like solvers in DPS, agents in MAS must share knowledge about tasks and partial works. Conversely to the DPS approach however, they must also reason about the process of coordination among the agents. Coordination is central to multiagent systems, without it any benefits of interaction vanish and the group of agents quickly degenerates into a collection of individuals with a chaotic behavior. To produce coordinating behaviors in MAS, most research has concentrated on developing groups in which both control and data are distributed. Distributed control means that agents are autonomous (to some degree) in their actions. Evidently, this autonomy can lead to uncoordinated activities because of the uncertainty of each agent's actions. In this context, a number of coordination techniques have been deployed. However, no technique investigated the relation between uncertainty and the situation addressed by agents. Indeed, the uncertainty decreases when the degree of familiarity of the addressed situation increases.

Our work presented in this paper is a step toward remedying this problem by providing a framework for designing multiagent systems in which agents are capable of coordinating their activities in routine, familiar and unfamiliar situations. We begin in Section 2 by motivating our framework. We then outline basic elements of our framework relative to three levels of cognitive control in Section 3. Section 4 details our mode of knowledge representation, and Section 5 explains how the proposed model can be used in multiagent systems. Section 6 provides some details on our implementation and presents results of our experiments. Finally, Section 7 concludes with some open problems.

2 Coordination between Agents: Guiding Principles

Our work has been motivated by our efforts to coordinate intelligent agents in domains like air traffic control [8] or urban traffic. The framework presented in this paper reflects an effort that has extended over several years. In this section we summarize the guiding principles which have led us to develop this framework.

2.1 Coordination is easier in Routine than in Unfamiliar Situations

In MAS, agents must find an *appropriate technique for working together in harmony* [20]. In fact, if all agents had complete knowledge of the goals, actions and interactions of their members, it would be possible to know exactly what each agent is doing at the present moment and what it is intending to do in the future. In this context, it would be possible to avoid conflicting and redundant efforts, agents could be perfectly coordinated and the effort of achieving this state would not be prohibitively high.

However, such complete knowledge about actual actions and reactions is only feasible in routine situations. In real-world domains, there are also familiar and unfamiliar situations. In familiar situations, agents can generally coordinate their behaviors since individual acts are carried out under expectations of future actions of other agents' actions and beliefs. In unfamiliar situations however, the coordination between agents is difficult to obtain and maintain because agents need to be constantly informed of all developments in order to elaborate their decisions. In fact, a complete analysis to determine the detailed activities of each agent is impractical in unfamiliar situations, and agents should have the capability to reason about others.

In multiagent systems, we are therefore interested in three kinds of interactions between agents: interaction in routine situations, interaction in familiar situations and interactions in unfamiliar situations. For these categories of interactions, the coordination between agents increases from unfamiliar to routine situations, whereas communication increases from routine to unfamiliar situations (Figure 1a).

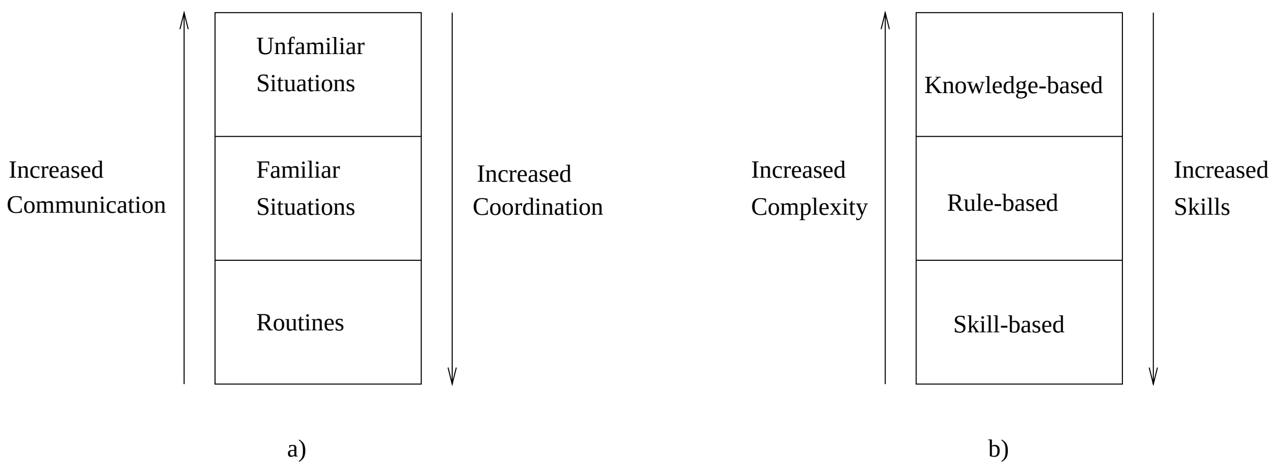

Figure 1: a) Coordination in different situations in a multiagent environment; b) an hierarchical model of human behavior and reasoning techniques.

A goal of our research is developing an architecture of agents with the conceptual models to investigate the three kinds of interactions. Generally, conceptual models have a hierarchical structure defined by the skill-rule-knowledge (S-R-K) levels (Figure 2b) of Rasmussen [24]. In the S-R-K perspective, the skill-based level denotes almost routines performances. At this level, agent performance is governed by stored patterns of predefined procedures, that map directly from observation (i.e. perception) to an action. The rule-based level represents more conscious behavior when handling familiar situations. The rule-based behavior is conventionally described by a set of heuristics, that is by a set of stocked rules. The knowledge-based level accounts for unfamiliar situations for which know-how or rules are not available. Indeed, for these situations the control of performance must move to a higher conceptual level, in which behavior is controlled by goal and utility and more generally by the reasoning about others.

2.2 Leading Agents to Prefer their Lower Cognitive Levels by the Integration of Social Laws

As previously noted, the coordination between agents generally decreases from routines to unfamiliar situations. More precisely, when agents have routine and familiar behaviors, these behaviors are generally known by all agents. In this context, any agent has facilities to coordinate its activities with other agents, and communication is requested only when necessary. In order to strengthen the levels relative to routines and familiar situations, we enrich each agent with social regularities (for instance: coordinative rules, cooperative rules, etc.) and social collectivities (e.g. roles, groups, organizations, etc.) in the form of social laws. By doing this, we assume that the agents adopt these social laws and each agent obeys these laws and will be able to assume that all others will as well.

3 An Agent Model based on Hierarchical Model of Human Behavior

3.1 Overview

It is becoming widely accepted that neither purely reactive nor purely planning systems are capable of producing the range of behaviors required by intelligent agents in a dynamic, unpredictable multiagent environment. Indeed, in these environments, agents require skills to respond quickly to familiar situations or routines, while simultaneously being able to carry out unfamiliar situations such as conflicts. Furthermore, in multiagent environments, an unfamiliar situation for an agent can be a familiar situation for another, and the former can request the latter to carry it out. Therefore, agents in complex, real-world domains need to combine the benefits of reactive and planning systems to control their behaviors. Recently, some approaches try to integrate these two levels [9, 13, 14, 23]. However, these approaches still seem incomplete since they do not incorporate the decision-making process that is important in multiagent environments [29]. Therefore, our agent model combines advantages of reactive, planning, and decision-making systems. Precisely, the proposed model in this work has been influenced by the skills, rules, and knowledge (S-R-K) levels of Rasmussen [24].

The skills, rules and knowledge-based processing proposed by Rasmussen reflects differences in consistency of response and conscious control of human behavior. Skill-based behavior refers to fully automated activities such as tracking or guiding, rule-based behavior to stereotyped actions such as test point checking in troubleshooting electronic circuit, and knowledge-based behavior to conscious activities involving problem solving or decision-making. We believe that this differentiation between the three cognitive levels is also applicable for multiagent environments where it is important to analyze the behavior of many agents with reference to their cognitive levels. Furthermore, we should concentrate on developing groups of agents in which both control and data are distributed. Distributed control means that agents are autonomous to some degree in their actions. This autonomy can however lead to uncoordinated activities because of the uncertainty of each agent's actions. To reduce this uncertainty, agents should have the propensity for skill-based and rule-based behaviors rather than knowledge-based behavior.

These considerations have led us to adopt Rasmussen's conceptual model as a framework to develop an agent architecture that evolves in a world inhabited by other agents. This model is driven by the goal of combining the complementary advantages of reactive, planning and decision-making systems in order to take into account different situations which arise in multiagent environments: routines, familiar and unfamiliar situations. First, it needs to be reactive to be able to quickly respond to changes in its environment. Secondly, it should be capable to plan its activities for a recognized task or goal. Finally, the model must also allow reasoning about others since agents should be capable of making decisions that take into account their own intentions and also others' intentions.

The proposed model (Figure 2) has been developed from the analysis of human behavior and includes the following phases. First, perceived information from the environment leads the agent to execute an action if the corresponding situation is perceived in terms of reflex action. If this is not the case, the agent tries to recognize the situation. It can recognize the considered situation in terms of an action or in terms of a goal (or a task). In the first case, it tries to execute the corresponding action, and in the second case it invokes the planning module. Finally, if the agent faces an ambiguity and cannot come to a decision, or faces many alternatives, then it invokes the decision-making module to make a decision in order to commit to achieve a goal or an action. A goal leads an agent to plan, that is to produce a sequence of actions that achieve the chosen goal.

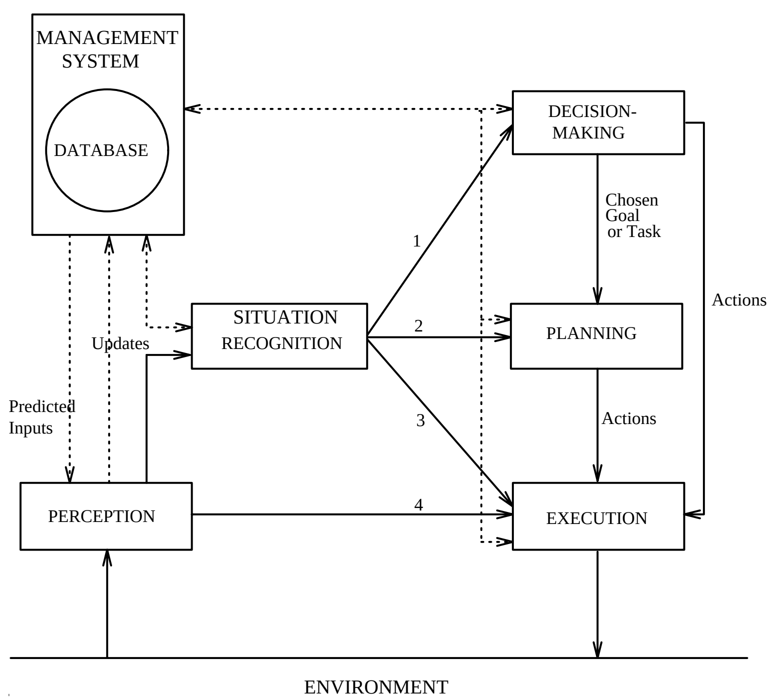

Figure 2: An agent model based on S-R-K framework—1 corresponds to an unfamiliar situation that needs decision-making; 2 is a situation recognized in terms of a goal; 3 is a situation recognized in terms of an action; 4 is a situation perceived in terms of reflex action.

Table 1. The relationship between the proposed model and the three Levels of control of human behavior.

Knowledge	perception - recognition - decision - planning - execution perception - recognition - decision - execution perception - recognition - planning[a] - execution
Rules	perception - recognition - planning[b] - execution perception - recognition - execution
Skills	perception - execution

[a] the planning process adapts old cases to the new situation, and the adaptation is significant.
[b] the planning process adapts old cases to the new situation, and the adaptation is generally minor

3.2 Towards an Hierarchical Model which Facilitates the Coordination between Agents

The model shown in Fig. 2 is actually made up of five knowledge processing activities and a global knowledge base including a database and a management system. In this subsection, we describe each processing activity and the global knowledge base.

3.2.1 Perception and Execution Modules

The perception and execution modules are aimed at providing the agent model with the necessary input-output capabilities. Input capabilities include: 1) sensing any entities which are initially unknown in the surrounding environment (for instance, other agents, obstacles, etc.) and, 2) receiving messages from other agents. Output capabilities include: 1) acting, that is effecting some actions (for instance, changing orientation, turning left, etc.) and, 2) sending messages to other agents in a a multiagent environment. In fact, the general function of the perception module is to extract information about entities, events, states and relationships in the external world, so as keep the world model accurate and up to date [2]. This module has also a specific function that consists of perceiving and reacting if the information is perceived in terms of action. To accomplish this, the perception module is organized into three layers: a sensing layer, an anticipation layer and a command layer.

Sensing layer: this layer is responsible for the perception of the current situation. In the real world, this layer monitors peripheral sensors, translates and filters data according to some pre-including software, and sends the results to the anticipation layer and to the database for updating. Notice that the term "situation" is used in a general way. A situation can be a snapshot of the outside world, as in [13], or a formal description of the world, as in [1], for instance. The idea is that a situation describes a subset of the world at a certain time.

Anticipation layer: we feel that being reactive implies an element of anticipation. To react appropriately to sudden, unexpected situations, you must quickly anticipate the course of events and act in a way that will change the anticipated issue. For instance, if you are driving a car and suddenly, a dog crosses the street in front of you, you're going to put the breaks on or try to avoid the dog, if your *anticipated* trajectory meets the dog's estimated trajectory. It is not only the view of the dog that suggests a radical action to you: seeing a dog cross the street in your rear-view mirror won't have the same effect. What causes your reaction is an anticipated collision between your car and the dog. The task of the anticipation layer will then be to anticipate the upcoming events (by processing the predicted inputs) and inform the command layer. Of course, the anticipation at this level can't be an exhaustive prediction of all possible outcomes. The anticipation has to be focused on some specific features. Furthermore, the computations involved in the anticipation process must not be too complex, so as to avoid abusive use of CPU time. After all, only a rough anticipation is required at this level. For example, the anticipation layer for an automated vehicle could involve the estimation of the trajectory of moving objects and the estimation of the vehicle's own trajectory to avoid collisions.

Command layer: sensing and anticipating lead to a *shallow* description of the actual situation: at this point, the agent perceives some entities around it and expects some potential consequences of its actions. If this description needs a reflex action, the command layer determines this action and sends it to the execution module. To determine a reflex action, the command layer compares the situation's description with stored procedures reflecting skill behavior. In this way, the direct link between the command layer and the execution module is a direct stimulus-response link with no representations, no conscious planning or decision-making at all, and which is completely automatic and completely situated specific.

Notice that there are several arrows going into the execution module (see Fig. 2) and consequently, this can lead to conflicts. To resolve this kind of conflict, our execution module associates a priority to each demand of execution according to a predefined interrupt vector, and executes this demand according to its priority.

3.2.2 The Situation Recognition Module

Generally, a situation pattern is a an element of the agent's model of the world. At each time interval, and if the current situation does not correspond to a reflex action, the situation recognition module is informed by the command layer of the perception module that a current situation needs recognition. Then, the situation recognition module tries to match the incoming frame, filled with information from sensory data, with predefined patterns of situations included in the world model. Precisely, the recognition module compares properties and attributes of predefined situations to corresponding properties and attributes of the considered frame. This comparison continues until a match is found. Reason [25] calls this process "similarity matching". If several patterns are candidate for a perceived situation, than the most frequent pattern is selected first [25]. In fact, this process is similar to "conflict resolution" in production systems.

Each situation pattern in the real world has an objective towards this situation is directed. This objective can be an action to be executed or a goal to be achieved. If the objective is ambiguous, the situation is considered as unfamiliar. Thus, a situation can be recognized in terms of action, in terms of goal, or finally as an unfamiliar situation. Then, the recognition module passes the control to the execution module if it is an action, to the planning module if it is a goal, or to the decision making module if the situation is unfamiliar.

3.2.3 Planning Process

In our framework, the planning process uses the case-based planning which is based on the idea of planning by remembering [19]. In fact, we view the process as a processus of remembering one or a small set of concrete instances or cases from familiar situations, and adapt it for the new situation. This new situation might be a familiar situation if the adaptation is minor (in this case the agent has just recall the plans that have worked before and make use of it directly), or an unfamiliar situation if the adaptation is significant (here, the agent recalls past operations from similar situations and modifies his behavior to suit the new situation). Memory-based planning or planning from cases means remembering failures so that they can be avoided, remembering successes so that they can be reused, and remembering repairs so that they can be re-applied. To achieve this, past planning experiences are organized in memory by two sorts of indexes: goals to be satisfied and failures to avoid. By organizing plans around goals as well as planning failures, an agent can avoid problems it has encountered before. In multiagent environments, these failures can be produced by the planner or by any another agent.

In our model, the case-based planning process consists of the following steps: 1) retrieve appropriate plans (or cases) from memory; 2) select the most appropriate plan(s) from those retrieved; 3) adapt the most appropriate plan to the new situation; 4) evaluate the adapted plan; 5) based on evaluation, modify and/or repair the adapted plan if it is not acceptable, else store the new plan in memory.

3.2.4 The Decision-Making Module

Sometimes, an agent will not be able to identify the information from the environment which act as cues for its decisions. In this situation, the agent cannot continually weigh his competing goals and concomitant beliefs, in deciding what to do next. At some point the agent must just settle on a state of affairs for which to aim. Deciding what to do establishes a limited form of *commitment*.

In our work, purposes of the decision-making module are: 1) to choose between alternative goals (because agents evolve in a multiagent environment); 2) to choose between alternative actions; 3) to choose between alternative plans. Notice that the decision-making process considered here is a distributed process where several agents coexist with their goals, actions, plans and utilities and where each agent is responsible for some portion of the decision-making effort. The approach adopted here uses a cognitive map (or causal graph) [3, 17, 36], that is a specific way of representing an agent's causal assertions about some limited domain. Specifically, a cognitive map is designed to capture the structure of the agent's causal assertions and to generate the consequences that follow from this structure. This map has only two basic types of elements: concepts and causal beliefs. The concepts are treated as variables, and causal beliefs are treated as relationships between variables. In multiagent systems, the goals, the plans and the utilities can be considered as concepts. These concepts might be the concepts of the decision maker or the concepts of another agent about which the decision maker is reasoning. Notice that the concepts of goals and utilities might be the concepts of a group or an organization about which the decision maker has to reason in order to decide.

The second type of basic element in a cognitive map is a causal assertion. Causal assertions are regarded as relating variables to each other, as in the assertion that "the amount of fuel of agent3 promotes its ability to cooperate for task4". Here the causal variable is "the amount of fuel of agent3" and the effect variable is "the ability of agent3 to cooperate for task4". The relationship between these two variables is indicated by the word "promotes". A relationship can be "positive" (+), "negative" (−), "neutral" (0), "neutral or negative" (i.e., nonpositive) ($\ominus$), "neutral or positive" (i.e., nonnegative) ($\oplus$), "nonneutral"($\pm$), "positive, neutral or negative" (i.e., universal) (u).

A. Calculating the indirect effect and the total effects

Once the reasoning process about relations is terminated, and the relationships between all of the variables are determined, the cognitive map can be drawn. Relationships that are in sequence form paths, and paths transmit indirect effects. For example, suppose there is a positive arrow from concept i to concept j and another positive arrow from j to k, then there is a path from i to k through j, and this path carries an indirect positive effect. The positive value of the arrow from i to j is combined with the positive value of the arrow from j to k to yield the positive value of the path from i to k through j. The operation of combining direct effects of relationships that are in sequence into indirect effects of a path is called *multiplication*. Precisely, the rules of multiplication are [3]:

1. $+$ *times* anything is that thing;
2. 0 *times* anything is 0;
3. $-$ *times* $-$ is $+$;
4. multiplication is symmetric. For instance, $+$ *times* $- = -$ *times* $+$.

When two or more paths start with the same point (i.e., concept) and end with the same point, their effects can be added into a total effect of the first point on the second. The operation is called *addition*. Similarly, the rules governing this operation are [3]:

1. 0 *plus* anything is that thing;
2. $+$ or $-$ *plus* itself is itself;
3. $+$ *plus* $-$ is $\pm$
4. addition is symmetric. For instance, $+$ *plus* $- = -$ *plus* $+$.

B. Solving the distributed decision-making problem

Generally, the cognitive map that represents the subset of a decision maker's belief system relevant to reasoning in a multiagent environment is converted to the form of a *valency* matrix V. This matrix is a square matrix of size n, where n is the number of concepts in the corresponding cognitive map. Each element v_{ij} characterizes the relationships between elements i and j. The valency matrix has a number of useful properties [15]. With the valency matrix, we can calculate indirect paths of length 2, 3, 4, etc. Matrices relative to these indirect paths are:

$$V_{ij}^2 = \sum_k v_{ik} v_{kj}$$

$$V_{ij}^3 = \sum_k v_{ik}^2 v_{kj}$$

etc.

For V^2, each of the terms of the form $V_{ik}V_{kj}$ expresses the indirect effect of a path from i to some k and from that k to j. Summing the effects of all such paths (through each possible concept variable k) gives the indirect effect of all paths of length 2 from i to j. Likewise, raising the valency matrix to the third power gives the indirect effect of all paths of length three from i to j. Raising the valency effects matrix to the q^{th} power gives the indirect effect of all paths of length q from i to j. In an acyclic cognitive map of n concept variables, there is no path longer than $n-1$. Therefore, the total effect matrix T, which has as its ij^{th} entry, the total effect of i on j for an acyclic cognitive map can be calculated, from the direct effects matrix with the operations *multiplication* and *addition* defined above, as follows:

$$T = \sum_{i=1}^{n-1} V^i$$

Now it is important to say how to solve the decision-making problem. Generally, given a cognitive map with one or more decision variables and a *utility* variable, which decision should be chosen and which should be rejected? To achieve this, the concerned agent should calculate the total effect of each decision on the utility variable. Those decisions that have a positive total effect on utility should be chosen, and decisions that have a negative total effect should be rejected. Decisions with a nonnegative total effect should not be rejected, decisions with a nonpositive total effect should not be accepted. Decisions with a zero total effect on utility do not matter. No advice can be given about decisions with a universal total effect or a non-zero total effect. Finally, we use heuristics about how to decide, and preferences between concepts to determine the final solution.

This total effect matrix can be used for generating advice based on the total effect of each goal, action or strategy on the utility variable. More generally, the elements of T may be used to guide a dynamic decision process until one goal is reached. Based on the calculation of T, we have the possibility of solving:

1. problems of a given cognitive map like this one: "could *concept*$_i$ be strengthened if *concept*$_j$ is strengthened?", "could *concept*$_i$ be weakened if *concept*$_j$ is strengthened?", etc.;
2. the problem of changes which can be formulated by: if certain relations change, what will happen to the considered cognitive map?
3. the explanation problem which consists of finding consistent explanations with the observed changes.

3.2.5 Database and Management System

The database and the management system allow an agent to make an estimation of the state of the world and to update its knowledge. The database can contain information about time, space, entities (including other agents), events and states of the external world. It also includes information about the agent itself, such as capabilities, motives, goals, strategies, preferences, specific knowledge about the application domain, structural and functional knowledge, physical laws, etc. Other knowledge may also be learned and modified during the process, particularly for the needs of case-based reasoning. In this category are included the successful plans, the plan repair strategies, the plan modification rules, the similarity metrics, etc.

The management system is an active module that stores and retrieves informations. It also contains a prediction capability that generates pertinent information for other modules, and particularly for the perception module. Finally, it also assures coherence and coordination between the activities of the layers.

4 Knowledge Representation

In our context, the mode of knowledge representation that we need should have the following characteristics. It should represent an appropriate way to formalize decision situation descriptions such as routine, familiar and unfamiliar situations. It should also make abstraction easy in order to take into account high level information such as what, how, when, who, etc., because such information allows agents to improve their coordination [12]. Finally, the knowledge representation adopted should make case-base planning possible in order to reflect adaptation from old cases to new cases, particularly from familiar situations to unfamiliar situations. In other words, the knowledge representation should reflect a dynamic memory structure that can change its organization with new experiences.

The theory of dynamic memory structures was first proposed by Schank and his team [30] to deal with problems in natural language understanding. Scenes, MOPs (memory organization packets), and TOPs (thematic organization points) are three kinds of high level structures that are used by a system with dynamic memory capabilities to represent and process information. Schank and his coworkers defined a scene as:

> "... A memory structure that groups together actions with a shared goal, that occurred at the same time. It provides a sequence of general actions. Specific memories are stored in scenes, indexed with respect to how they differ from the general action in the scene."

The concept of MOP has been introduced as a way to structure and index scenes. Schank gives the following definition for a MOP:

> "A MOP consists of a set of scenes directed towards the achievement of a goal. A MOP always has one major scene whose goal is the essence or purpose of the events organized by the MOPs."

A MOP is used to represent knowledge about classes of complex events. For instance, "going to movie theater" would be represented with a MOP. A MOP contains a set of *norms* which represent the basic features of the MOP, e.g., what goals are achieved, what events occur, it who is involved, etc. Thus, the norms relative to "going to movie theater" would include: there is an human agent who intends to go to see a movie in some place. MOPs that are more specific versions are specializations of some other MOPs that are abstractions . Thus, "going to movie theater" and "going to theatre" are two MOPs that are specializations of the " going to entertainment" that is an abstraction of "going to movie theater" and "going to theatre". Notice that any MOP that refers to an occurrence of a particular event is an *instance*. Finally, MOPs are joined together by links that a reasoner follows to get from one MOP to another. In this way, a reasoner can determine *what* information is available and *when*.

The second kind of high level structure that is used for dynamic memory are the TOPs. These structures are independent of a particular domain, and are used to characterize abstractly the interactions between different contexts. More precisely, TOPs represent knowledge about how goals can interact. Thus, they can reflect "side effect", "prevention", etc, that exist between goals.

It is clear that the concept "scenes" is very close to our notion of "situation" and consequently we use it to modelize this notion. The concept of MOP has been introduced as a way to structure and index scenes. Therefore, we use MOPs to adequately index our situations. Precisely, we adopt a model of knowledge representation which is similar to the model of Riesbeck and Schank [27]. In this model, there are basically two kind of MOPs: instances and abstractions. We use names starting with `M.` for abstractions and names starting with `I.` for instances. Instances sit at the bottom of the abstraction hierarchy. They have abstractions but no specializations. Instances represent cases, individual events, or objects. Abstractions are generalized versions of instances or other abstractions. One MOP is an "immediate abstraction" of another if there is a direct link from the more specific MOP to the more abstract MOP. Normally we will be interested in all the abstractions of a MOP. These include the MOP itself, its immediate abstractions, the immediate abstractions of the immediate abstraction, etc.

The central process in MOP-based memory is the one that searches memory. The basic idea starts with a MOP and a set of slots describing an instance of that MOP. Each slot has a packaging link, called a *role*, and the MOP, called the *filler* that the link points to. The goal consists of searching for the most specific specializations of the MOP that have slots compatible with the input slots.

Finally, we use TOPs for causal relations (+, $-$, 0, etc.) between agents' goals, plans and utilities.

5 The Proposed Model in a Multiagent Environment

We will now study interaction between agents more specifically. We start with the contribution of social laws to coordination between agents.

5.1 Integrating Social Laws into the Proposed Architecture: A Contribution to the Coordination Between Agents

As previously mentioned in section 2, it is clear that we need to enrich the low levels of our model. By doing this, agents might have behaviors (routine and familiar behaviors) that are known by each of them. Thus, any agent has facilities to coordinate its activities with other agents, and negotiation (and more generally communication) is requested only when necessary. In order to enrich the low levels, we insert, in each database, social regularities and social collectivities in the form of social laws. By doing this, we assume that the agents adopt these social laws and that each agent obeys these laws and will be able to assume that all others will as well [31].

Now, we detail our social laws with examples concerning the urban traffic application to get a feeling for how we really implement these laws. The discussion that follows was adapted and expanded from appendix of [4].

Social regularities embody norms and rules. Social *norms* are expectations shared by group members which specify behaviors that are appropriate for a given situation. Note that expectations are both anticipatory and normative in nature, and thus norms considered here are also. Rules are used to designate coordinative rules, cooperative rules, collective rules and regulations. *Coordinative rules* are social rules like driving on the right, speaking in turn, etc., whose point is to coordinate the activities of a number of agents who are trying to do more or less the same thing with a minimum of interference from others. Each agent who follows such a rule is doing his part in a joint effort to coordinate the activities for everybody involved. *Cooperative rules* reflect behavior rules whose performance of an action by one agent makes sense only if many others do the same thing: recycling cans and papers, consuming energy, etc. These cumulative actions yield collective goods or prevent collective evils. Collective goods are goods that everyone enjoys if everybody collaborates, such as a clean environment, etc. Collective evils hurt everyone as is the case with air pollution for example. *Collective rules* organize collective actions out of individual efforts. Rules that divide labor in an organization, a tribe or a company are collective in this sense. Finally, *regulations* are standards that are enacted, promulgated, or otherwise imposed on a group by some mutually recognized authority. This is the case for standards imposed by a policeman on drivers at a crossroads.

On the other hand, social *collectivities* can be distinguished by the degree of structure: roles, groups and organizations. The term role is highly ambiguous and can refer to a role category, to the expectations associated with that category (role expectation), or to the expected behavior itself. We restrict the term *role* to the expected behavior and we use the term *position* to designate a place in a group or a social relation. Thus for example, the role of a policeman at a crossroads indicates an expected behavior that all drivers know. *Groups* on the other hand refer to a collection of agents mutually regarding themselves as a unit (we) to which they belong, united by having something in common, including a mutual interest that is presumably furthered in their group activities. For example, in urban traffic, cars constituting a convoy is a group of cars. Finally, a formal *organization* is much more structured than a group. Its members are clearly differentiated by position to which specific duties and responsibilities are attached. Furthermore, collective rules prescribe interconnecting roles that organize the activity of agents in the same and different positions. Unlike in a group, in an organization the members need not have anything in common (other than being members) or any common interest, and they need not share any "we" feeling. For example, in urban traffic, we might considered the organization of policemen or the organization of motorists.

Note finally that according to what we observe in human societies, the social laws are generally learned and not integrated. In this paper however, we adopt the "integration" of these laws in our architecture in order to design a "social agent" and show how these laws influence the coordination between agents. The learning of social laws is very difficult to achieve and we leave it as theme of our perspectives.

5.2 Using the Proposed Model in Multiagent Environments

In a multiagent environment, interactions between agents can lead to: 1) a structure reflecting interaction between decision-maker(s) and actor(s), 2) a structure where the coordination is done by a knowledgeable agent and 3) a structure without any hierarchy. In this section, we show how these structures are constructed with our model.

5.2.1 Interaction between Decision-Maker(s) and Actor(s)

In some multiagent environments, an agent higher up the hierarchy may have a vast array of sensors, much more complex processing (it "knows" more situations than other agents and it has the ability to recognize these situations), and a whole regiment of agents to effect the actions upon which he decides. Our model of Fig. 2 can be used recursively in this type of structure, as shown in Fig. 3, to reflect interaction between decision-maker(s) and actor(s).

As shown in Fig. 3, an agent$_i$ perceives a vast array E and elaborates some desired state that it communicates to another agent$_j$ (under its responsibility). The desired state can be elaborated at any level (skills, rules or knowledge) of agent$_i$ depending on its experience and on the considered situation. In turn, agent$_j$ evaluates the desired state at its knowledge-based level. This evaluation can lead it to *negotiate*, to *refuse* or to *act* according to the hierarchical links between it and agent$_i$. In the case where it decides to act, the agent$_j$ acts on an environment under control of agent$_i$. By this way, it communicates to agent$_i$ what it is doing by signals through the perception module. Notice that in the case where agent$_j$ must execute what agent$_i$ says without any reclamation, it must only act at the skill level (whereas the agent$_i$ has the possibility to reason at the three levels: skill, rule and knowledge).

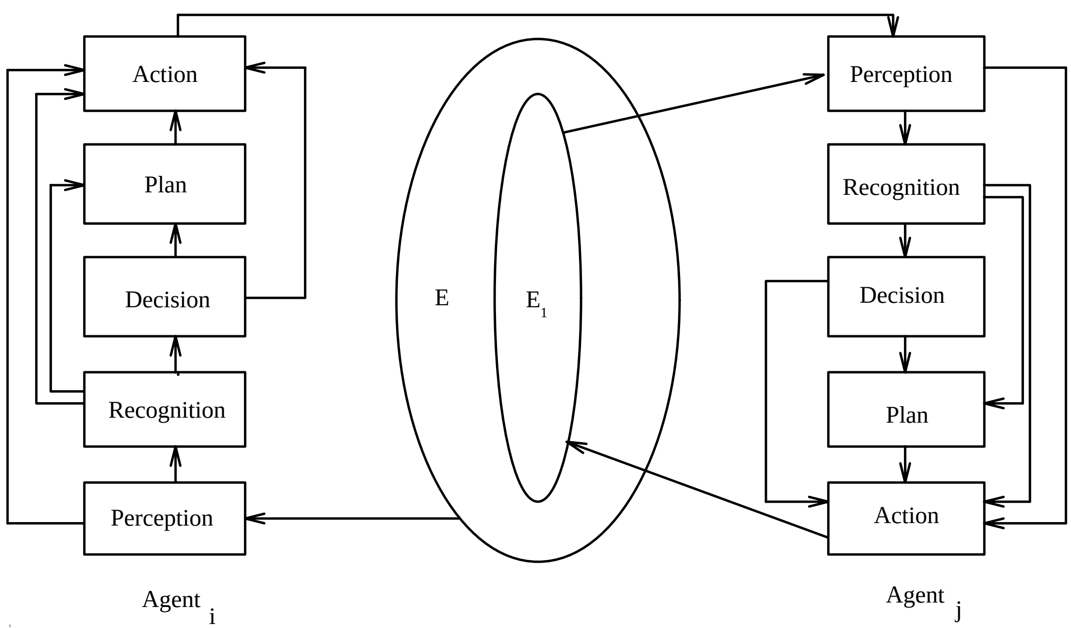

Figure 3: Interaction between a decision-maker and an actor.

5.2.2 Coordination by a Knowledgeable Agent

In multiagent systems, one might have a knowledgeable agent which have several subordinates, whose environments may overlap as for instance a policeman and drivers using the same space to drive in urban traffic. Fig. 4 depicts such a situation, and makes it evident that an important consideration for the knowledgeable agent (agent$_k$) is to avoid setting goals for his subordinates which may lead them to a conflict. More precisely, agent$_k$ has the responsibility to assure the coordination between agent$_i$ and agent$_j$. To achieve this coordination, it must specify to the subordinates abstract behaviors like *what, when, who* and *where* in order to prevent negative relations [33]. These abstract behaviors are generally conceived at knowledge-based level. Notice that if *what* specifies actions to do, agent$_i$ and agent$_j$ act to skill level and coordinate the execution of actions by the mutual knowledge about *when, who* and *where*. If on the other hand, *what* reflects goals to achieve, the two agents plan these goals taking into account the other abstract information. Finally, it is useful to note that reports on evolving actions done by agent$_i$ and agent$_j$ are communicated to agent$_k$ through the perception (i.e. by signals [6]).

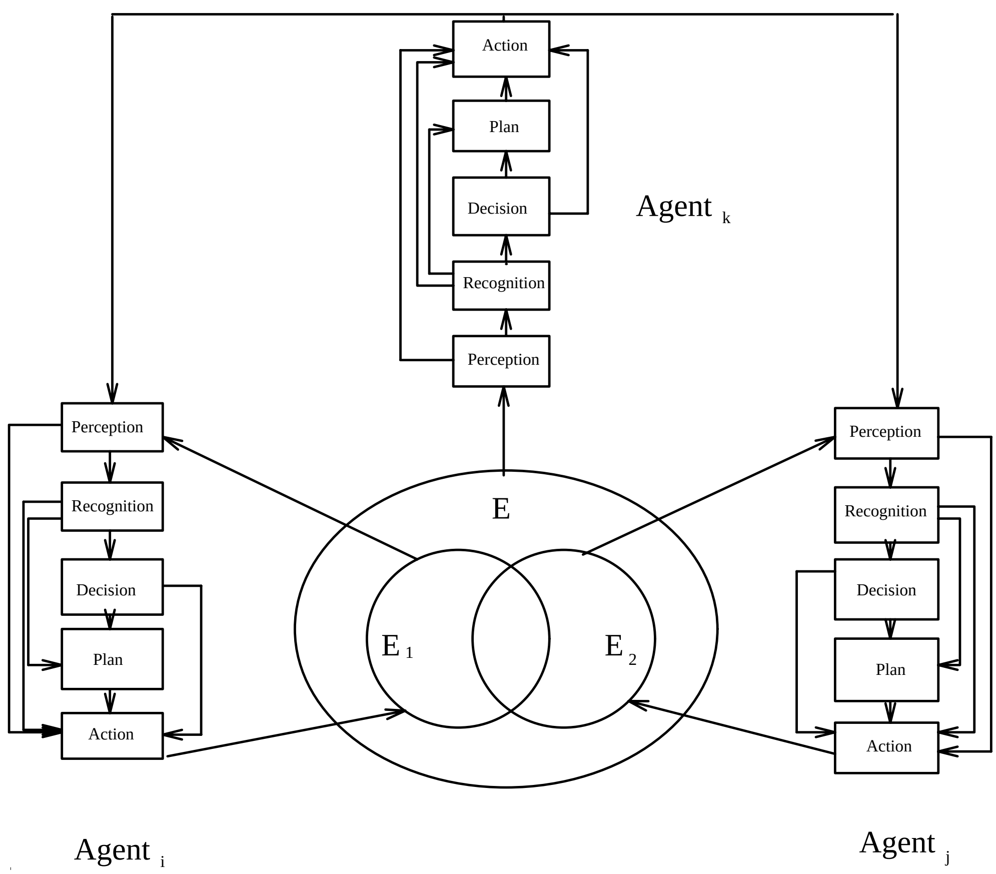

Figure 4: Coordination by a knowledgeable agent.

5.2.3 A Non-Hierarchical Structure

In a non-hierarchical structure (Fig. 5), there are four cases to consider: a) each agent has his environment for perception and action, and the environments are disconnected, b) There is one environment for perception in which there are disconnected environments for actions, c) there is one environment for perception in which there are connected environments for actions, d) each agent has his environment for perception which is also his environment for actions, and the environments are connected.

In case **a)**, if actions and plans are not done for a global task or for a global goal, and if there are no resources to share between agents, then agents do not need to coordinate their actions and their plans. In this context, each agent has to act or to achieve his goal without any interaction with others. If on the other hand, agents seek to achieve a global task (or goal) or if they share resources, then agents have to coordinate their behaviors. In this case, the coordination is very difficult to obtain and to maintain since agents do not share any portion of environment.

In case **b)**, $agent_i$ perceives what action $agent_j$ is doing and vice versa. Therefore agents i and j communicate through the perception-action modules. This communication by signals facilitates coordination between agents. Interaction through perception-action modules also allows agents to make *mutual adjustment*, where each actor makes on going adjustments to manage the interdependencies [21]. If agents need more coordination, they can communicate more explicitly with signs and/or symbols by using a specific language for communication. Notice that agents' activities are not space-related since E_1 and E_2 are not connected. Consequently, we can only have temporal relationships between agents' activities that agents have to manage as interdependencies (sequencing and synchronizing for example) in order to coordinate their activities. Notice that if agents' activities are completely independent, it does not make sense to refer to coordination between agents. Finally in this context, agents act and/or reason at skill, rule, or knowledge levels depending on their abilities and their know-how about the situations.

Case **c)** is similar to case b) concerning the communication by perceptions-actions modules. However in case c), environments E_1 and E_2 are connected and consequently some agents' activities (which are included in E_1 and in E_2) are generally time-space related. These activities need more coordination and more communication than in the case b). For these activities, agents can make mutual adjustments in order to coordinate their activities. If this adjustment is not sufficient, they can communicate more explicitly by using a specific language. For those activities which are not included in E_1 and in E_2, case c) is similar to case b).

Finally, case **d)** has two parts. One concerns the intersection of E_1 and E_2 and is similar to its corresponding part in case c). The second part concerns the complement of the intersection and is similar to case a).

6 Example Implementation and Experiments

We now give details about an illustrative example and its implementation using the hierarchical architecture and the other concepts developed in this paper.

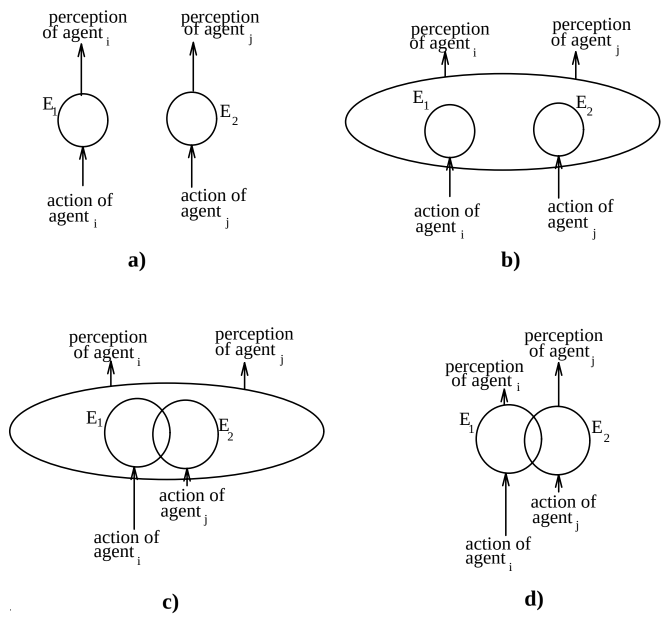

Figure 5: Perception and action in a non-hierarchical structure.

6.1 Urban Traffic: A Multiagent Domain

Urban traffic is a highly interactive task between various agents. These agents can be people (drivers, policemen, pedestrians, etc.) or machines (vehicles, traffic-lights, etc.) and have to continuously adjust their actions in order to avoid conflicts such as traffic-jams, and in severe cases, crashes.

In most situations of urban traffic, a human agent has to watch his environment and react appropriately in order to execute his action, or achieve his goal. In fact, each individual knows what to do and knows how to coordinate his activities with other agents. In most *routine* and *familiar situations*, this strategy is successful since human agents have developed appropriate behaviors in the face of arising conflicts. These appropriate behaviors are generally constituted by some learned behaviors plus some social behaviors due to social laws known by each agent. Thus, in most situations of urban traffic, human agents use only the low levels (i.e. skills and rules) to elaborate their behaviors.

Sometimes however, too many agents sharing space and time produce a traffic-jam or, in severe cases, a crash. Here, it seems that agents make use of the knowledge-based level in order to make decisions to solve the conflicts between them. Of course, if there is a policeman who makes decisions for all drivers, then the policeman makes use of his knowledge-based level to elaborate the decisions, and drivers execute them at the skill-based level or rule-based level. This case is a coordination by a knowledgeable agent.

Notice that communication in urban traffic is important since it allows agents to make mutual adjustments in order to coordinate their activities. In urban traffic however, agents communicate by signals and/or signs [6] and consequently the scope of this type of coordination is limited to the range of agents' perception of these signals and signs.

Thus, the urban traffic application is a relevant area of research for the multiagent paradigm. Consequently, we might demonstrate the applicability of the major concepts developed in this article for this application. However, due to the complexity of the problem domain it is not possible to give a complete validation in urban traffic. Hence, many aspects of our model cannot be investigated (for instance, the learning mechanism, the relationship between plan of agents, etc.). This section will, therefore, focus on investigation of global aspects such as interaction between agents in routines, familiar, and unfamiliar situations and leaving the others for the future.

More precisely, in this section we investigate a multiagent scenario showing traffic situations at an intersection (Fig. 6). This scenario captures: 1) agents' behaviors in routine and familiar situations (e.g. the agents' behaviors are coordinated by traffic lights); 2) agents' behaviors in complex situations (e.g. the agents' behaviors are coordinated by a policeman); 3) agents' behaviors in unfamiliar situations (e.g. the agents' behaviors are coordinated by social laws, because traffic lights are off).

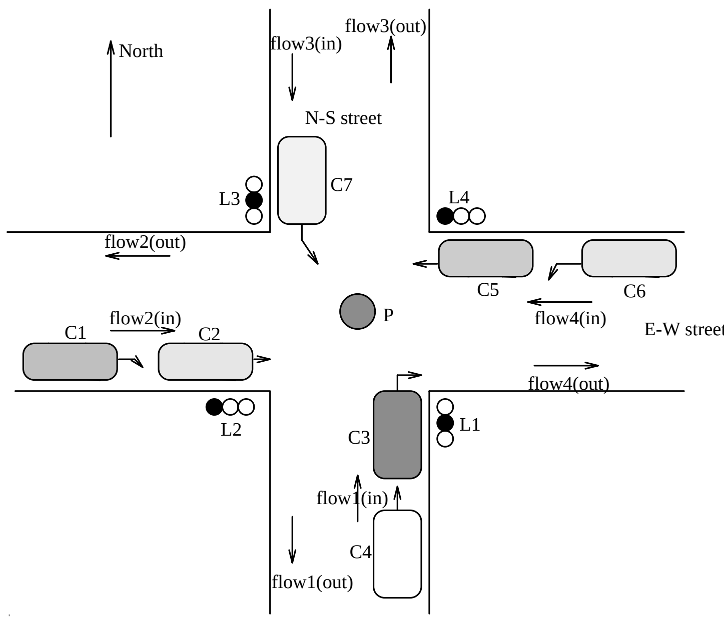

Figure 6: A multiagent scenario in urban traffic: the crossroads : C denotes a car, L traffic-lights, and P the place of a policeman.

6.2 Implementation

The architecture described in this paper depicts a general model and can be adapted to a vast number of fields. We implemented each component of the architecture (in Common Lisp) while keeping in mind this idea of generality. In order to validate this architecture in a multiagent environment, we have implemented the crossroads scenario using our architecture and the other concepts developed in this paper. Details of this implementation are given elsewhere [6].

6.3 Experiments

As stated previously, there are three levels (S-R-K: skills, rules and knowledge) of cognitive control in multiagent systems (MAS). These three levels can be grouped together into two general categories [26]. K is concerned with analytical problem solving based on symbolic representation, whereas S and R are concerned with perception and action. S and R levels can be only activated in routine and familiar situations because these low levels require that agents know the perceptual features of the environments and the knowledge relative to these situations. The K level, on the other hand, is only activated in unfamiliar situations. These considerations have been taken into account in designing our agents.

With these agents, we have developed implementations and experiments in urban traffic to verify our intuitions about the distinction between the two modes of processing: perceptual processing and analytical problem solving [34]. Perceptual processing is fast, effortless and is propitious for coordinated activities between agents, whereas analytical problem solving is slow, laborious and can lead to conflicts between agents. To this end, we are conducting a series of experimental studies on three policies of the crossroads scenario. The policy 1 refers to a routine of urban traffic. In this routine, agents' activities are coordinated by traffic lights. Policy 3 refers to an unfamiliar situation of the crossroads scenario. In this situation, agents should rely on social laws to make decisions because traffic lights are off, and there is no policeman to coordinate their activities. Finally, the policy 2 refers to a complex situation where agents' activities are coordinated by a policeman, that is by a knowledgeable agent.

We examined for the cars three performance indices when comparing the policies: communication, processing time for each mode of reasoning (skills, rules and knowledge), and task effectiveness. The effectiveness is specified by two distinct parameters: errors and waiting time. A summary of the main experiments is given in Table 2. For the policeman who intervenes in the policy 2, we examined two performance indices: communications and processing time for each level of the agents' cognitive control. The results about these indices are given in Table 3.

Table 2. Experiments summary for the cars (these results are averaged across 10 scenarios.)

Performance\Policies	Policy 1	Policy 2	Policy 3
Communication (i)	1.9	3.1	5.3
Proc. time for S (ii)	140	145	30
Proc. time for R (iii)	17	130	17
Proc. time for K (iv)	6	60	420
Waiting time (v)	163	335	467
Errors (vi)	0.2	2.7	4.1

(i)mean messages (signs and/or symbols) sent per car while waiting to pass
(ii) mean Sun Sparc cpu seconds per car for reasoning at the skill level, while waiting to pass
(iii) mean Sun Sparc cpu seconds per car for reasoning at the rule level, while waiting to pass
(iv) mean Sun Sparc cpu seconds per car for reasoning at the knowledge level, while waiting to pass
(v) mean Sun Sparc cpu seconds per car for the total waiting time at crossroads
(vi) mean number of near misses or collisions for all cars in a scenario

Table 3. Experiments Summary for the Policeman (Parameters in this table are measured during the total waiting time per car.)

Performance\Policies	Policy 2
Communications	8.7
Proc. time for S	40
Proc. time for R	90
Proc. time for K	205

As we had anticipated, our implementation and experiments successfully demonstrated that perceptual processing is fast, effortless and is propitious for coordinated activities between agents. Policy 1 is in this case, since it is considered (in our implementation) as routine of urban traffic.

More precisely, *policy 1* which reflects a routine, performed best overall. Particularly, this policy allows agents to have the best waiting time and there is no effort since the processing time at the skill level is much higher than at other levels. As we had anticipated, our implementation and experiments successfully demonstrated that with a routine, the number of near misses or collisions between cars is the fewest. In addition, the number of messages sent per car in this routine is also the fewest. Consequently, our expectation that coordination at the skill-based level is more easy to obtain and to maintain than at knowledge-based level, is proved true. In the same context, our expectation that routines are not generally communication intensive is also proved true. What happens in this routine is that agents share social laws that allow them to respect traffic lights, and therefore to coordinate their activities without communicating intensively.

The *policy 3* is considered (in our implementation) as an unfamiliar situation where agents reason at the knowledge level to elaborate their decisions. In this context, we had also expected that the analytical problem solving, that is the knowledge-based level, would be slow and laborious. These expectations are proved true as indicated by our results. Precisely, policy 3 performed worst overall. Indeed, agents have the worst waiting time and their major processing time is at the knowledge-based level where reasoning is about itself and about others. This reasoning mode is considered as requiring effort since it needs knowledge about others' intentions in order to predict dimensions relative to actions or plans such as for instance what (to do) and who (is the actor). By doing this, agents try to improve their coordination. Finally, with the policy 3, the number of near misses or collisions between cars is the greatest. This result is also in accordance with what we had expected, that is that the analytical level is more propitious to errors and therefore to poor coordination between agents.

Finally, the *policy 2* is a complex situation where cars are coordinated by a knowledgeable agent who is the policeman. In this type of situation, we had expected that cars reason at low levels and the policeman at a high level. Our implementation and experiments confirm this expectation. Precisely, processing times in Table 2 indicate that agents consider the situation relative to policy 3 as a situation which is more familiar than policy 3 but less routine than policy 1. In the same context, processing times in Table 3 show that the same situation is considered by the policeman as an unfamiliar situation with a certain degree of familiarity. These considerations led in the context of cars, to a policy which is intermediate in performance as shown in Table 2, since the waiting time of policy 2 is intermediate between the waiting times of policies 1 and 3.

Our implementation and experiments demonstrated also that the presence of a policeman improves the coordination between cars since the number of near misses or collisions is lower than the number of near misses of the policy 3 where there is no coordinator. The number of messages sent per car is also lower than the number of messages of policy 3 since cars in policy 2 only follow indications given by the policeman. However, implementation and experiments show that the coordination in the case of policy 2 is not as efficient as the coordination in the case of policy 1.

Collectively, our implementation and results are consistent with what we had expected.

7 Open Problems

As stated previously, some aspects have been implemented and experimented and some others are left for the future. These leaving aspects tie into open problems and can be summarized by: How can we ensure consistency between agents? How do we manage the relationship between the plans of agents? What is the impact of communication on case reasoning? How can agents coordinate their next near-term behaviors while they carry out their current activities? More generally, how can agents make predictions about their next near-term behaviors at low levels? How can agents efficiently learn (long-term research)? More precisely, how the control of agent activity shifts from the the knowledge-based level through the rule-based level to the the skill-based level in the case of self-instruction and from the rule-based level to skill-based level when an agent-instructor is active? Finally, how can agents learn social laws?

8 Conclusion

The framework described in this paper was motivated by a problem: how to permit agents to coordinate their activities in routine, familiar and unfamiliar situations. The first step taken toward solving this problem was to determine what agent model was associated with these situations. Common sense revealed that coordination is generally more easy to obtain and maintain in routine than in unfamiliar situations. Conversely, unfamiliar events must face the problem of reasoning about others and consequently agents which are in this situation can use communication intensively if they do not succeed in making a decision about what to do next with other agents. In addition, for unfamiliar situations, agents are more propitious to make negative interactions. In these conditions, a viable approach to agent design for multiagent systems must be able to combine benefits of reflexive, planning and decision-making systems in order to produce skill-based behaviors, rule-based behaviors and goal-oriented behaviors as required by real multiagent environments. To this end, we have proposed an architecture for agents that reflects three levels of cognitive control as specified by Rasmussen's taxonomy: a skill-based behavior, heuristic-based behavior, and knowledge-based behavior. The next step taken was to enhance the proposed architecture so that agents prefer low levels, that is the skill and rule levels. In this respect, we argued that agents should be provided with social laws which allow them to rely on low levels since these levels are easier for efficient interactions. In the remainder of the paper, we showed how our proposed model can be used in multiagent environments. Finally, our implementation and experimentation was consistent with what we had expected, in particular: perceptual processing is fast, effortless and is propitious for coordinated activities between agents, whereas analytical problem solving is slow, laborious and can lead to conflicts between agents.

References

[1] P. Agre and D. Chapman, Pengui: an implementation of a theory of activity, *Proc. of AAAI-87*, Seattle WA (1987) 268-272.

[2] J. S. Albus, Outline for a theory of intelligence, *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-21 (1991) 473-509.

[3] R. Axelrod (ed.), *Structure of Decision: The Cognitive Maps of Political Elites*, Princeton University Press (1976).

[4] K. Bach and R. M. Harnish, *Linguistic Communication and Speech Acts*, MIT Press, (1979).

[5] A. Bond and L. Gasser, eds., *Reading in Distributed Artificial Intelligence*, San Mateo, CA, Morgan Kaufman (1988).

[6] B. Chaib-draa, Coordination between agents in routine, familiar, and unfamiliar situations. Rapport de recherche DIUL-RR-9401 du Département d'Informatique, Univ. Laval, Ste-Foy, PQ, Canada (1994).

[7] B. Chaib-draa, B. Moulin, R. Mandiau et P. Millot, "Trends in distributed artificial intelligence," *Artificial Intelligence Review*, vol. 6 (1992) 35-66.

[8] B. Chaib-draa, and P. Millot, A framework for cooperative work: an approach based on the intentionality," *Int. Jour. of AI in Engineering*, vol. 5 (1990) 199-205.

[9] P. R. Cohen. M. L. Greenberg, D. H. Hart and A. E. Howe, Trial by fire: understanding the design requirements for agents in complex environments, *AI Magazine*, vol. 10 (1989) 34-28.

[10] S. E. Conry, K. Kuwabara, V. R. Lesser, R. A. Meyer, Multistage negotiation for distributed constraint satisfaction, *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-21, (1991) 1462-1477.

[11] E. H. Durfee, *Coordination of Distributed Problem Solvers.* Kluwer Academic Publishers, Boston, (1988).

[12] E. H. Durfee and T. A. Montgomery, Coordination as distributed search in a hierarchical behavior space, *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-21 (1991) 1363-1378.

[13] I. A. Ferguson, Touring-machines: autonomous agents with attitudes, *IEEE Computer*, vol. 25 (1992).

[14] J. R. Firby, An investigation into reactive planning in complex domains, *Proc. of AAAI-87*, Seattle, WA (1987) 202-206.

[15] F. Harary, R. Z. Norman, and D. Cartwright, *Structural Models: An Introduction to the Theory of Directed Graphs*, New York: Wiley (1965).

[16] N. R. Jennings, *Cooperation in Industrial Multi-Agent Systems* world Scientific Series in Computer Science, vol. 43 (1994).

[17] B. Kosko, Fuzzy cognitive maps, *Int. Jour. of Man-Machine Studies*, 24 (1986) 65-75.

[18] S. Kraus and J. Wilkenfeld, Negotiations over time in a multiagent environment: preliminary report, *Proc. IJCAI-91*, Sydney, Australia (1991) 56-61.

[19] J. L. Kolodner, An introduction to case-based reasoning, *Artificial Intelligence Review*, 6(1) (1992) 3-34.

[20] T. W. Malone, Toward an interdisciplinary theory of coordination, CCS TR #120, MSA Center for Coord. Sc., MIT, Camb., MA (1991).

[21] H. Mintzberg, *The Structuring of Organizations.* Englewood Cliffs, NJ: Prentice-Hall (1979).

[22] T. M. Mitchell, Becoming Increasingly Reactive, *Proc. AAAI-90*, Boston, MA, (1990) 1051-1058.

[23] J. P. Müller and M. Pishel, An architecture for dynamically interacting agents, *Int. Jour. of Intelligent and Cooperative Systems*, 3(1), (1994), 25-45.

[24] J. Rasmussen. *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*, North Holland (1986).

[25] J. Reason, Cognitive aids in process environments: prostheses or tools?, *Cognitive Engineering in Complex Dynamic Worlds*, E. Hollnagel, G. Mancini, and D. d. Woods, eds., Academic Press, London (1988).

[26] J. Reason, *Human Error.* Cambridge, UK: Cambridge Univ. Press (1990).

[27] C. K. Riesbeck and R. C. Schank, *Inside case-based reasoning*, Lawrence Elbaum associates, Publishers, Hillsdale, New Jersey (1989).

[28] J. S. Rosenschein, Consenting agents: negotiation mechanisms for multiagent systems, *Proc. IJCAI-93*, Chambéry, France, Aug. (1993) 792-799.

[29] A. P. Sage, Information systems engineering for distributed decisionmaking, *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17 (1987) 920-936.

[30] R. C. Schank, *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*, Cambridge University Press (1982).

[31] Y. Shoham and M. Tennenholtz, On the synthesis of useful social laws for artificial agent societies, *Proc. of AAAI–92*, San Jose, CA, USA (1992) 276–28.

[32] K. P. Sycara, Argumentation: planning other agents' plans, *Proc.IJCAI-89*, Detroit, Michigan, Aug. (1989) 517-523.

[33] F. von Martial, *Coordinating plans of autonomous agents*, Lecture Notes in Artificial Intelligence, Springer-Verlag Series, Berlin (1992).

[34] K. J. Vicente and J. Rasmussen, Ecological interface design: theoretical foundations,*IEEE Trans. Syst., Man, Cybern.*, vol. SMC-22 (1992) 589-606.

[35] M. Wooldridge, and Jennings, N. R., Agent theories, architectures, and languages: a survey, in *Agent Theories, Architectures, and Languages* eds. M. Wooldridge and Jennings N. R., LNAI 890, Springer Verlag (1995), 1-39.

[36] W. R. Zhang, S. S. Chen, and R. S. King, A cognitive map based approach to the coordination of distributed cooperative agents, *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-22 (1992) 103-113.