Integration of Generic Motivations in Social Hybrid Agent

Fenintsoa Andriamasinoro and Remy Courdier

IREMIA, University of La Reunion, BP 7151, Messag Cedex,
97715 Saint-Denis de La Réunion, France
{fenintsoa.andriamasinoro, remy.courdier}@univ-reunion.fr

Abstract. Most hybrid agent architectures are constructed with a hierarchical succession of reactive (at a lower level) and cognitive (at a higher level) layers. Each of these layers represents a behavior, a function, a decision, etc. Instead of using such functional layers, we propose in this paper a generic model of a social hybrid agent, which is rather based on natural (human/animal) motivations of the agent. We discuss here the contribution of our approach in hybrid agent modeling. The present work uses the American psychologist Abraham Maslow's pyramid of needs. The basis of this modeling is then using the result of an existing real-world psychological study.

1 Introduction

1.1 Human and animal behavior

In agent modeling, animals are considered as reactive entities, behaving according to their internal and external impulse and the dynamic of the environment, be it in an individual or in a social context [4], [9]. On the other hand, humans are regarded as cognitive entities which can evaluate the actions to be performed (we call this a high-level behavior), according to many parameters "imposed" by their society (policy, role, etc.) [11]. However, this high-level behavior in a human being is actually the realization of his animal instinct but performed in a more rational form (intentional coordination, etc.). For example, when a hungry dog finds food in a kitchen, it immediately eats it (we call this a low-level behavior). On the other hand, a person first asks to whom the food belongs, and if he can eat it, he first takes a plate and a fork, etc. In any case, this person's behavior aims to satisfy the animal instinct in him, which is the hunger. The combination of both behavior levels forms the hybridism.
Besides the social norm, the human also makes, for instance, a high-level organization: the need for eating is manifested at a higher-level by the need for everybody to work. The remuneration resulting from this work will then be used for buying foods, satisfying then the need to eat (where *eating()* is the low-level behavior).

We may see from the above concepts that humans and animals have common basic motivations: the satisfaction of natural needs (hunger, sexual impulse, etc.)

but the two categories differ in the way they satisfy them. This concept is also used in agent modeling, [3], [4]. These motivations (also called source of actions by [7]) make an agent behave either reactively or cognitively (i.e. adopting a low or a high level behavior). As we see, the concept of *motivation* takes an important place in the study of agent behavior.

The objective of the work presented in this paper is to integrate this theory of basic motivations in hybrid agent paradigm, by taking into account these two levels of behavior. The idea here is to propose a generic model that integrates abstract motivations (also called "abstract common source" of actions) which may then be instantiated according to the studied application. For this purpose, we use Abraham Maslow's pyramid of needs [2], [3], [18].The basis of this modeling thus uses the result of an existing real-world psychological study.

Because we model the human/animal behavior and we also use the pyramid of Maslow (initially based on human needs), our work then concerns the modeling of social agents, those having the need to live in a society [11] (also see Section 4). This work integrates the concept of hybrid agents [23] that will have needs, feelings[(1)], and so on, during their activities.

The terms *needs* and *motivations* presented in this paper may be confusing. Actually, the basic motivation of an agent corresponds to the satisfaction of its (or others) basic needs (see Section 3.2). However, as they are in fact equivalent, we will adopt the idea that these two terms can be alternatively used, depending on the context of our explanation.

1.2 Modeling motivations in hybrid agents: state of art

Hybrid agent architectures Most hybrid architectures are constructed of layers, each of them defining a specific function and possibly a decision. We may mention TOURINGMACHINES [8] a model having three layers (from bottom to top): the reactive, the planning and the modeling layer. Another layered architecture is INTERRAP [15]. In this approach, each successive layer represents components: a behavior-based, a plan-based and a cooperation-based component, the overall is connected in a knowledge base. More recent layered architectures are ICAGENT [19] which models the intention reconciliation and planning in agents, and GLA [17] in which layers regroup similar types of computations instead of a functional decomposition as in the previous models.

The common characteristic of these layered architectures is that the layer components generally define behavior, plans, cooperation, decision, etc. We also note that the lowest layer of most of them integrates the reactive part of the model (because it is close to the environment), followed at a higher layer level by the cognitive part (and possibly other additional layers).

Relation between motivation and actions The layers in the architectures presented above model actions rather than their source: the agent motivations

[1] The notion of feelings will be studied in a future work but the present paper outlines our idea for doing it.

[7]. If we consider the hybridism concept from the angle of a "combination" of reactive and cognitive models, the notion of motivations may be found, particularly at the reactive level. At this level, the motivation (satisfying hunger, avoiding obstacles, etc.) is the main factor that determines the behavior of the agents, followed by an action selection process [9], [16]. In this case, we consider the motivation as explicit. The above examples are however chosen depending on the application (ants, robots, etc.).

On the other hand, in works about cognitive modeling, the motivation concept is less considered even if the dynamic of the environment (leading to a reactive/instinctive behavior) is taken into account [14], [22]. Works about cognitive modeling are more focused on organization, coordination, etc. However, some works use the notion of *desire*, particularly those using the BDI framework [5], in which a *desire* is a goal driven by a motivation. But even in these cases, the notion of motivations is not very clear and finally, considered as *kept* by the agent designers' mind during the design.

1.3 Objective and issues

In brief, the modeling of motivation in a hybrid agent is either not considered at all or considered depending on application. In this paper, we propose a generic hybrid agent model called MASLOW (acronym for Multi-Agent System based on LOW needs) from the same name as the psychologist, but also especially based on Low-Needs concepts (Section 3.3). By integrating this generic model, we aim to overcome what we consider to be a "lack of motivation" concept in hybrid agent modeling.

The problem related to this work is the balancing between the reactive and cognitive agent behavior when it is known that an agent has a goal-directed behavior when it has to cope with the environments[2] (e.g. [13]). In addition, and given that we base our work on motivations, another issue is to determine the degree of motivations for the agent for satisfying each need of the pyramid. The one having the higher degree is treated first. In our current study, this problem is oriented towards the determination of the need semantically being the most important. Criteria must be given. Obviously, the importance of needs may vary from time to time and then, a repetitive process of checking this importance must be performed. Furthermore, because the agents in the model are social agents, we must consider the way in which they should balance between individual and social contexts.

1.4 Preamble

In our work, the cognitive part of the agents is not yet studied deeply. However, we plan to integrate it in the future (see Section 9.2). Meanwhile, many assumptions are first made to deal temporarily with this situation. We assume

[2] The environment of the agent may be the internal one, such as impulse, etc., or physical one or the other agents in the system

that cognitive agents already have a plan (see Section 5.1 about this notion) and each cognitive agent has knowledge of the need state of others. We agree that in real-world case, this latter assumption is not realistic because in a general way, the knowledge of an agent about its environment is partial [7], [10]. Thus, at this stage of the work, it may represent a limit of our model and requires further investigation. However, it does not really affect the modeling of our hybridism concept, that is, the balancing between the reactive and the cognitive behavior of our agents (Section 5.2).

MASLOW is currently developed in JAVA. Thus, some notations in this paper also follow the syntax of this language.

The remainder of this paper is organized as follows: Section 2 presents an overview of the case-study we analyze throughout this paper, followed by the basic concept of our model: the description of needs (Section 3). Next, Section 4 particularly presents our concept of social needs according to both the pyramid definition and the agent modeling. After presenting these concepts of needs, we outline in Section 5 their relation to actions. All of these parameters are necessary before we can define in Section 6 the criteria needed in the management of the importance of the need. The whole model is evaluated in Section 7 and analyzed in Section 8. Lastly, Section 9 concludes the paper and gives our perspectives of the work.

2 Case study

Our current case study concerns two docker agents D_1 and D_2 working in a harbor and paid daily. They have to carry heavy goods from storage to a boat (storage→boat=1 journey). In the storage, there are many goods but the daily work consists of carrying just 8 of them (only 1 unit per journey is possible). The adopted coordination at cognitive level is that D_1 will carry 3 goods whereas D_2, 5 goods, both following a road, and assuming that at the road's edges, there are "dangerous ravines" beyond which, there is the sea (Figure 1). As the harbor is a dynamic environment (ex: containing grounded obstacles, a crane which may inadvertently "release" something, etc.), D_1 and D_2 must consider this dynamic when performing their work.

Let us also assume that after the docker carriage repartition is made (respectively 3 and 5 goods), each of them commits himself to fulfilling his respective task.

The initial cognitive plan P of the docker is:
P:={goTo<storage>, take<good>, goTo<boat>, put<good>}
until the 8 (=3+5) ores are carried. This plan is then the high-level behavior of the dockers.

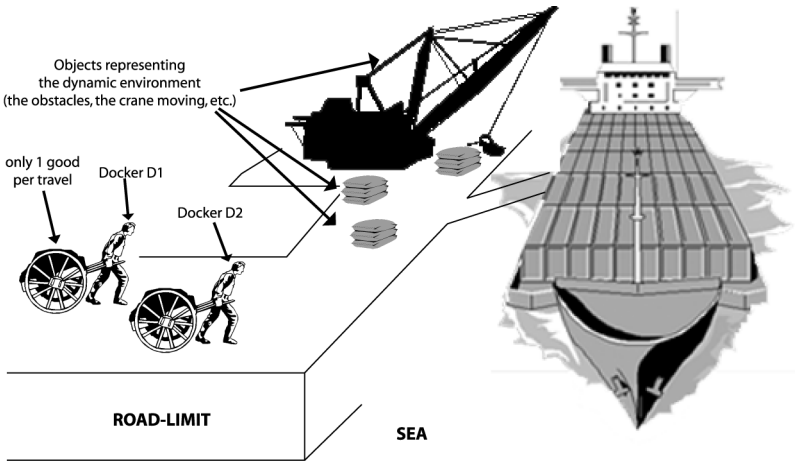

Fig. 1. The scheme of the case study. The two dockers D_1 and D_2 have the cognitive plan to carry goods but in a dynamic and dangerous environment.

3 Basic concepts

3.1 The pyramid of Abraham Maslow

Our present work uses the Pyramid of MASLOW (hence noted Π) of the American psychologist Abraham Maslow [18]. Π regroups the five hierarchical needs of a Human Being: physiological needs, the need for security, the need for love, the need for esteem and the need for self-realization. This last level is not yet analyzed deeply in this work

According to this psychological study, *all actions led in the Living Being's behavior are motivated by at least one of these five hierarchical needs.* We call them *abstract* needs as each species of Living Beings has its own (or sometimes common) way to satisfy them. But the final objective is the same: to satisfy one or another of these basic needs. As the needs are abstract, terms such as need for *security* or *social* actually corresponds to any needs which are set at these levels, which are then not always unique.

The architecture of the pyramid itself is one of our reasons for choosing this model in our work (additional reasons are found in [3]). Indeed, it results from a real-world psychological study of human behavior (like biologists study animal behavior). In our agent modeling, each level then has a specific conceptual semantic not based on pure hypothesis.

3.2 The pyramidal need

Formalization The fine-grain need of Π is called PN for Pyramidal Need. A formalization of PN was presented in [3]. Generally, it corresponds to the need state: the *quiet (sufficient),* the *threatened (limit)* and the *missing (insufficient)*

ones. However, this formalization does not take into account the state where the need is "over satisfied" (in an excessive way) while it actually may occur in many real situations. An example of this last state is a person who is fed too much (the need 'hunger' is over satisfied) or who has high blood pressure, etc. We then first introduce this state in this paper (see Figure 2). Note that only the *sufficient* and the *insufficient* states are obligatory. The other states may be omitted, depending on the semantic of the PN.

A PN is normally written $PN_{level/rank}$ in which *level* and *rank* are the indexes (like a "physical" place) of the need in Π. But in our formalization, these two parameters may be omitted if not necessary (just write PN). Note that the *level* parameter is considered during the determination of the degree of motivation of a PN while the *rank* one is not (see Section 6 for more precisions).

The states are presented throughout an axis on which a cursor - the indicator of the current state - slides. We call its current position *ccpos*. The zone corresponding to the *sufficient* state is called the *ideal zone* of which the middle position is called *mizpos* for middle ideal zone position. If *ccpos*==*mizpos*, the need is fully satisfied. Each PN need has a unit called *PNu* (e.g. liters, meters, pounds, etc. depending on the application).

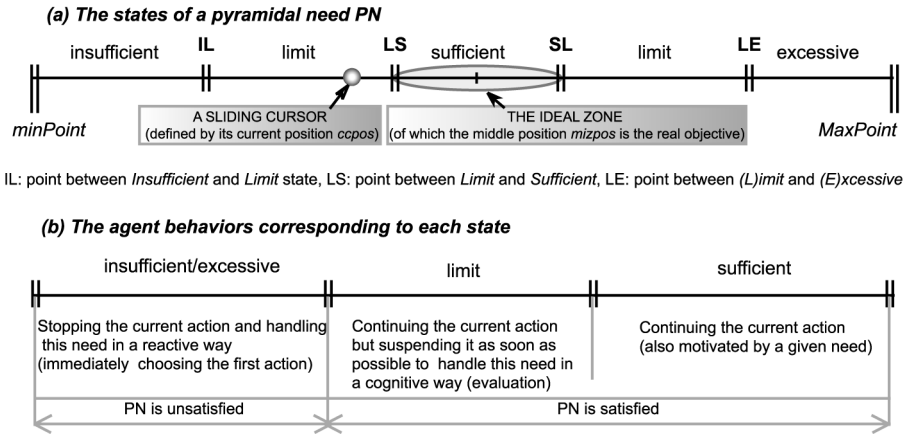

Fig. 2. The formalization of a Pyramidal Need (PN)

The formalization of a state is the following:
- agent.PN.xxx:= expr_1 (comp_op.) expr_2 [(logical_op.) expr_1 (comp) expr_2...]
- expr_i := sub_expr_1 [(arithmetical_op.) sub_expr_2 ...]
- sub_expr_1:= *variable* | *constant*

In which:
- xxx := insufficient | excessive | limit | sufficient *//one of the possible need state*
- comp_op:= $> | < | \geq | \leq$
- logical_op:=$\wedge|\vee$
- arithmetical_op:= $+ | - | * | /$

- variable:= any dynamic values // *variables are of any type, and may or may not be one of the agent properties. They may be returned by a method.*
- constant:= IL | LS | SL | LE | other fixed values // *Constants may also be returned by a method. IL, LS, SL, LE are the points which separates states* (Figure 2).

Note: The most important thing to remember is that each sub_expr_i must return a *numeric value* so that the principle of the representation by the cursor system holds.

As example about D_2, the transport of the goods is formalized as follows:
$D_2.work.insufficient := D_2.nbGoodsCarried < 5$,
$D_2.work.excessive := D_2.nbGoodsCarried > 5$
$\Rightarrow D_2.work.unsatisfied := D_2.nbGoodsCarried < 5 \vee D_2.agent.nbGoodsCarried > 5$
in which *work* is the need PN for the agent to finish the required work.

Concretely, this example means that D_2 will not feel satisfaction until the 5 goods he is committed to carrying are carried. But D_2 also feels the same sensation if he thinks of carrying beyond 5 goods (the state of *work* is going to the *excessive* state), the above formalization meaning that there is a given reason for not doing so (e.g. no more wage even if conveying more goods, or, no more place in the boat to stock them, or, the boat cannot support more than 8 goods, etc.).

3.3 Low/High Needs (LN/HN)

The Basic-Needs (BN) and High-Needs (HN) [3] are the only possible type of PN. We here rename BN to LN for Low-Needs, due to confusion we met with the notion of "basic", also used to specify the basic level of the pyramid (i.e. the level 1). As a need PN is either LN or HN, these two notations constitute the *type* of PN. In sum, {PN}={LN}∪{HN}.

The LN corresponds to the "inborn" needs of the agents, also called the *natural parts* of the agents: hunger, sleep, preservation instinct, etc. The LN is *permanent* and is always *active* (considered in behavior). The LN-area (see left side of Π in Figure 3) is similar to what the psychologists call the *collective unconscious* [6]. On the other hand, the HN corresponds to the needs that (only) cognitive agents have, resulting from its plan, intention, reasoning, etc. HN is temporary and is active or not. Indeed, in general, high-level goals differ from one agent to another (even if in our case study, they are the same).

What we emphasize is that *one HN is always motivated by at least one LN* (see Equation(1)). Symmetrically, one LN may be the motivation of more than one HN. *LN* is an impulse to be satisfied and *HN* is the cognitive adopted goal (in form of desire) to directly or indirectly satisfy *LN*. The docker D_1 has for instance the desire *HN* with *HN.sufficient:= nbGoodsCarried==3* to be satisfied. But the satisfaction of this need is actually motivated by the satisfaction of a *LN: =satisfy_hunger*, that is, the wage from the work will be used to buy some food.

This relation between HN and LN is formalized via a functional relation $f \in F$, joining the high and low level needs (F designates the set of these

functions). The function f may be first a composition of other functions. In other words, it may happen that $f = f_1 o ... of_n$, $with\ \overline{1..n} \in \mathrm{N}$

$$\forall L_N,\, H_N \in \Pi,\, \exists f \in F / L_N = f(H_N) \tag{1}$$

Obviously, the inverse of Equation (1) is not always true (i.e. f is not surjective). The *LNs* do not always have a corresponding *HN* because an agent has not always to perform a high-level behavior in order to satisfy a low-level need. Actually, the functions f_x are actions.

Figure 3 summarizes our description of needs. All needs are managed by a module called the Need Importance Manager (NIM), which gives the present most important need requiring priority treatment (see Section 6).

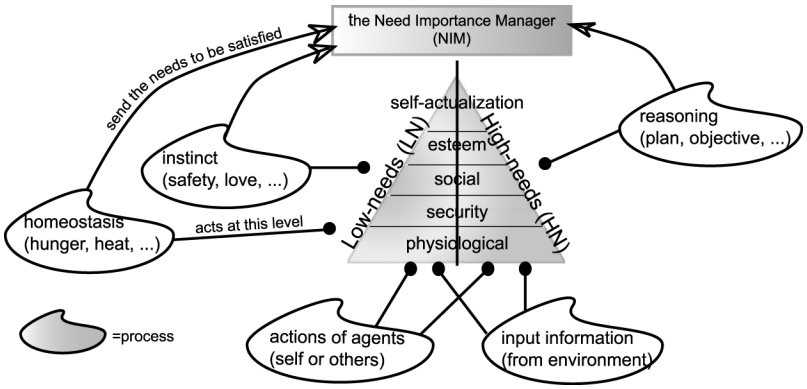

Fig. 3. The hybrid architecture based on need managements

3.4 The need variation speed (PNVS).

The PNVS is the speed at which the cursor of a PN is sliding on the axis. It is an important parameter as it determines the approximate time for the agent to react. For instance in the case where an important PN is worsening, the more the PNVS is elevated, the more agent "feels" to quickly treat this PN. Note that the only source of a PN-cursor moving (if any) is internal or external actions (i.e. from the agent or from the system).

The PNVS is evaluated as follows: there is an initial time t_0 which is initialized, either at the first time the PN is created in the agent, or the last time the PNcursor in the axis has changed direction. Let x_0 the cursor-position at t_0. The value of PNVS is determined by Equation (2). Note that in the MASLOW system, the position of the cursor is considered every δ *timeunit*.

$$PNVS = \frac{ccpos - x_0}{|current\_time - t_0|} * \frac{ccpos - mizpos}{|ccpos - mizpos|}(PNu/timeunit)\,. \tag{2}$$

The operand on the right only aims to get the sign of the current direction of the cursor, compared to the ideal zone. Then, if the resulting PNVS<0 (respectively>0, or ==0) then the state of PN is said *worsening* (respectively *improving*, or *stationary*). However, if between two times t and t+δ, the state moves from {*insufficient or excessive*} to *sufficient* (respectively from *sufficient* to {*insufficient* or *excessive*}), the state of PN is said: *very worsening* (respectively *very improving*).

3.5 Notations and definition

For a best comprehension of the remainder of the paper, the following notations must be set:

- predicates *PN.isxxx()* indicates that the description attributed to PN.xxx is verified, *xxx* being either one of the need state (e.g. PN.isSufficient()), or the evolution of the need state (e.g. PN.isWorsening()).
- the following relation is mentioned:
  insufficient/excessive < *limit* < *sufficient*
  (at the present period of our research, no comparison can be made between the insufficient and excessive state).
- predicates *isHN(PN)* and *isLN(PN)* respectively indicates if PN is a HN or a LN (see Section 3.3). Additionally, there is a function named *type_of(PN)* which returns the type of PN (LN or HN). Then, *isHN(PN)==true*$\Leftrightarrow$*type_of(PN)=HN*.
- when a need PN_1 is more important than another, PN_2, it is noted $PN_1 > PN_2$ (and meaning that PN_1 must be treated before PN_2).

In addition, in this paper, we call *the need checkpoint* the moment during which the Need Importance manager or NIM (see Figure 3) checks the state of all other needs in the pyramid and determines if there is or is not a more important need (than the current being treated) to be satisfied. Section 6 explains the way in which an important need is found.

4 The levels 3 and 4: the social needs

4.1 Principles

Like the individual needs, the social ones of Maslow (the abstract needs to be liked or to be esteemed) are also motivations of agents' behavior in social context.

We note the following basic needs:

- not being alone: this corresponds to the basic need to be in an environment where there is at least another congener (even if at this stage there is not yet any relationship between them). As [4] said, the social entity is in a society even if it is alone. This is an intrinsic characteristic of such entities.
- being integrated or being loved: this need is also valid for animals [4]. Example, the ones which integrate a group (to be liked), a natural chief of an animal society who wants to be respected (to be esteemed), etc. There is

also a manifestation of this need at human level: for instance, during their work (and actually, even beyond), the dockers want respect from each other (in way of talking, in policy, etc.).
- considering the others: particularly the impact of its action on others [12]. As far as possible, the agents generally do not take actions which worsen the need state of others. As a trivial example, let us take the *avoid()* action. The docker D_1 avoids D_2 when they meet during the carriage, not only due to his own preservation of instinct, but also due to the respect of the preserving instinct of D_2.

At a cognitive level, many acts can be mentioned when looking for love, esteem, consideration, etc.:
- the commitment [11] when working[3]: by committing to accomplish their works, the dockers want to show to others that they work perfectly. They also know that if they do not do so, they will be "rejected[4]" by their society (in [7], it is called a functional motivation). At the current stage of our research, the concept of being loved is limited to only this functional motivation (cf. the present case-study). But we agree that there are different "stages" of love for which each social entity is waiting: being loved by family, by friend, loved by colleagues, and so on. This interesting aspect is to be considered in a future.
- helping: helping is also a social act. But its realization depends on the other(s) agent(s) to be helped (congener? family? son? friend? etc.). The basic motivation for helping may then vary: the social obligation to working well (also a functional motivation), the search for a friendship in return for the help (=searching love), the search for esteem from the other, etc.
  Generally, other parameters must be taken into account: the degree of relationship between the two dockers, individual objective of each docker, etc. This generalization will be studied in a future work.
- accepting the influence of others by following their attitudes (this is also called a relational motivation by [7]). These attitudes may be motivated at a basic level by the search of love from other agents. One notable work tending to this influence is that of [20].

The formalization of love and/or integration is inspired from the work in [21] which discusses the utility function of the agent in its society, based on a social reward. The reward increases (i.e. the social need tends to be satisfied) if the agent is performing one more social action. We use this work because its idea can be adapted to the way in which we formalize our PN.

For instance, the following formalization may be given for PN:= *to_be_loved* (remind: a functional motivation):
love.sufficient:=work_not_done==0, *// the work, with regards to the docker's group, is the social action*

[3] We note that we consider here only the social level. But we agree that the commitment may also be made in the context of individual domain (self-commitment)

[4] At a basic level, being rejected is felt as not being loved any more.

love.limit:= work_not_done>6 ∧ work_not_done≤20,
love.insufficient:= work_not_done>20

When *love.insufficient* becomes true, the docker is excluded from his work (a situation personally felt as a "reject" from the group in which he is).

4.2 Balancing between individuality and sociality

The characteristic of the pyramid itself involves an implicit balancing between individual and social needs. In fact, the need management does not take into account the semantic (physiological, social, etc.) of each need. It considers only the state of all needs as well as their level number. Then, if the last important need is for instance at level 1 and the current treated need is at level 3, there was just an implicit balancing from the individual to the social need.

5 Needs and actions

5.1 Formalization of Actions

There are two kinds of actions:

- a *primitive*, the fine-grain action. It is uninterruptible when executed. Due to this characteristic, the need checkpoint is possible only between the execution of two consecutive primitives.
- a *plan*, composed by one or more primitives and intentionally prepared by cognitive agents. To be executed, a plan is recursively decomposed like a tree, until having the leaf (the primitives). By definition, a sub-plan is a part of a plan but situated at a lower-level in the tree.

The relation between needs and actions is set as follows: when executed for satisfying a need, an action α is repeated until a condition, called a *local_need* is satisfied. It is a PN locally related to a given action. Each action then has its associated *local_need*. For example, a docker who is going to the storage has the following parameters:
- the action α= *Goto(storage)*,
- having PN=local_need(α)/ PN.sufficient:= *self.position==storage.position*, *// note that here, PN is a HN because associated to a plan*
it means that the docker has a local (and psychological) need to reach the storage (i.e. want to have the same position as it). He will perform α until this need is satisfied.

5.2 From reactive to cognitive behaviors

To satisfy a given need, there is a list of n (n≥1) actions among which agents may choose. Choosing an action in a reactive way means that agent randomly takes one action between these n actions. Doing so in a cognitive way means that the agent evaluates each action. We thus use the Action Selection Mechanism (ASM)

[9]. Our criteria during the action selection process are presented in [3]. The balancing between the reactive and cognitive behavior depends on the current state of the need PN to be treated. When the PN-state is *insufficient/excessive*, the agent acts reactively. Otherwise, it acts cognitively, and by evaluating the appropriate action.

Acting in order to treat an unsatisfied need PN means suspending the current action the agent is performing. The suspension is first planned when the need PN to be treated is in a *limit* state. Furthermore, the suspension of the current action (let β) (also performed due to a previous unsatisfied need PN) can be possible, only between two primitives. If so, the model checks if PN is more important than PN'. In such a case, PN will be treated. Otherwise, the satisfaction of PN' via β is resumed.

6 Managing the need importance

6.1 Recall

The choice of the next action depends on determining first which need is the most important at the phase of checkpoint (assume, at the moment t). The need checkpoint (previously defined in Section 3.5) then actually consists of determining, for the time t, the degree of motivation for the agent to satisfying each PN (noted PNDM) of his Π. As the value of these PNDM may vary from time to time, the result of the checking at time t is then valid for only this time. This means that a need may not be important at a time t but may be so at *t+1*, depending on the agent activity.

If the checking issue results that $\mathrm{PN}_1 > \mathrm{PN}_2$ (PN_1, $PN_2 \in \Pi$), it means that the agent is more motivated to satisfy PN_1 than PN_2.

The degree of motivation for a PN depends on its type (HN or LN), his level in Π and his *state_ratio* (see Section 6.2, §*the urgency*). In other words,
PNDM=f(type_of(PN), level(PN), state_ratio(PN)).
This relation derives from the criteria we already proposed in [3]. The relation between PNDM, the type and the level is determined by the criteria recalled in Equation (3), in which *x, i, j* represents a level number.

$$\begin{array}{l} 0 - \forall PN_{i/rank},\, PN_{j/rank} \in \Pi \Rightarrow PN_{i/rank} > PN_{j/rank}\, \forall\, i < j \;\; (i,\, j \in \overline{1..5}) \\ 1 - \forall LN_{i/rank},\, HN_{i/rank} \in \Pi \Rightarrow LN_{i/rank} > HN_{i/rank} \;\; (i \in \overline{1..5}) \\ 2 - if\, \exists LN_{i/rank},\, HN_{x/rank} \in \Pi,\, \exists f_i \in F / HN_{x/rank} = f_i(LN_{i/rank}) \Rightarrow x := i \end{array} \tag{3}$$

6.2 New criteria

Resolution of a level classification problem The problem in Equation (3) is that Criterion 1 only compares HN and LN at the same level. When they are in a different one, it no longer holds, and it seems that Criterion 0 is better. However, if we strictly apply Criterion 0, the expression:

$\forall i < j, \in \overline{1..5}, \forall HN_{i/rank}, LN_{j/rank} \in \Pi, \Rightarrow HN_{i/rank} > LN_{j/rank}$
becomes true, and involves for instance that the high-need for transporting a good (a $\mathrm{HN}_{1/rank}$ motivated by the need to eat) is more important that avoiding an object falling down from the crane (a $\mathrm{LN}_{2/rank}$ need related to the security). Nevertheless, we know that this situation is not always realistic particularly when the need to be secure is threatened. Thus, in such a case, Criterion 0 may not be exceptionally applied. The new criterion described by Equation (4) is a proposed solution. It stipulates that if the state of a LN_i is *unsatisfied* (and only in this case), and the agent is performing a HN_j (with j<i), then the HN_j is suspended and LN_i is performed, *until it is led back to the limit state.*

$$\begin{array}{l} \forall x,\, i \in \overline{1...5},\, \forall PN_{x/rank},\, LN_{i/rank} \in \Pi \;/ \\ \quad if\, LN_i.isUnsatisfied() \\ \qquad * \;if\, isHN(PN_{x/rank}) \Rightarrow LN_{i/rank} > PN_{x/rank} \\ \qquad * \;if\, isLN(PN_{x/rank}) \;\Rightarrow\; LN_{i/rank} > PN_{x/rank}\, (but\, only\, if\, x > i) \\ \quad otherwise\, valid(criterion\, 0) \end{array} \tag{4}$$

Cloning Another principle is also introduced in this paper: cloning (see Equation(5)). In fact, it happens that one HN is motivated by more than one LN. Then, the HN is cloned as many times as the number of the corresponding needs LN. The interest of cloning is in the different "basic satisfaction" in which one given HN is involved. All cloned HN have the same structure but, once created, they then evolve differently.

$$\begin{array}{l} if\, \exists LN1,\, ...,\, LNk,\, HNx \in \Pi, \\ \qquad \exists f_1, ...,\, f_k \in F\, / HNx = f_1(LN1)\, , ..., HNx = f_k(LNk) \\ \Rightarrow create\, HNx1 := clone(HNx), ...,\; create\, HNxk := clone(HNx) \end{array} \tag{5}$$

The execution of the plan P is for example based on two basic motivations: the motivation for having something to eat, and the functional motivation, as mentioned in Section 3.4. The local need associated to P is then cloned and ranged respectively at level 1 (as a physiological need) and at level 3 (as a social need).

Classification of the local needs This is made as follows:

- let α, β two plans, α being a sub-plan of β, if (HN_α=local_need(α) and HN_β=local_need(β))$\Rightarrow HN_\beta > HN_\alpha$
- if a plan α is made by a series of actions (α_1, ..., α_i) $\Rightarrow$local_need(α_1) >...>local_need(α_i)

Urgency The notion of *urgency* is an additional criterion we first propose in this paper to resolve the case where, after applying both type and levels criteria, there are still two or more PN that have exactly the same importance. In fact,

in [3], the state_ratio[5] noted *sr* was already a first solution for this case, but *sr* is only a spatial criterion. As the need is temporally dynamic, a spatiotemporal criterion is better. For that, we then use the PNVS proposed in Section 3.4.

Assume that, the preserving instinct of D_1 leads him to have two motivations:

1. to avoid a grounded obstacle situated at 2,5 lengthunit from him (=a need PN_1 to be away from the closer grounded obstacles). The cursor moving characterizes the moving of D_1, and the minpoint of PN_1 corresponds to the position of the grounded obstacle. Here, the PN_1VS is the speed at which D_1 is walking.
2. to escape from an "aerial" object which is falling down from the crane, and going to fall directly onto him (=need PN_2 to be away from "aerial" obstacles). The currently falling object is situated at 8 lengthunit above D_1. The minpoint of PN_2 is the current position of D_1. The PN_2VS is the speed of the heaviness corresponding to the force of the gravity.

According to only the *sr* criterion, the most important need to be treated will be PN_1 (because 2,5 lu<8 lu). But when we take into account the PNVS, the situation is somewhat different. Indeed, it is sufficient that $2{,}5*PN_2VS > 8*PN_1VS$ so that PN_2 theoretically reaches its *minpoint* before PN_1 and in this case, the theory of *sr* is no longer valid. Actually, *sr* Criterion may be applied only if after the applying the urgency criterion, PN_1 and PN_2 still have the same importance. On the whole,

PNDM=f(type_of(PN), level(PN), PNVS, state_ratio(PN)).

To summarize, given *i* needs PN1, ..., PNi:

1. applying first PNDM= f(type_of(PN), level(PN)) as described in previous Subsections.
2. for the remaining PN needs (if any), comparing them by applying the principle of urgency (see Figure 4, Example 1 for illustration).
3. about the remaining PNs (also if any), considering the current state of each of them
4. using the state_ratio (Figure 4, Example 2 which shows another case).

7 Evaluation

7.1 Implementation

Our model is evaluated via a prototype. The implementation is organized in three main layers:

- a kernel layer, gathered under the maslow.kernel package and implements the classes of the concepts which are studied in the present work;
- an appli layer, found under the maslow.appli.docker package and corresponds top the implementation of the prototype;

[5] As a reminder, the *state_ratio* is the current relative position of the cursor in the axis, compared to the *minPoint* (see Figure 2 about the *minpoint*)

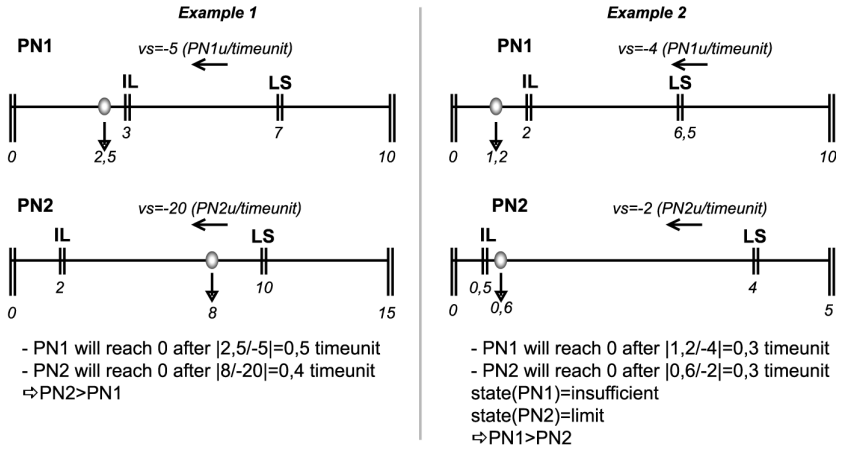

Fig. 4. Two examples of how to evaluate PN having the same importance according to the urgency and the states of the compared needs.

– a gui (Graphic User Interface) layer. This last one is not actually an interface designed for the Maslow model but is rather an adaptation of a generic GUI already used in a previous project (see [1] and also Figure 6. This GUI is connected to Maslow via the package maslow.appli.docker.gui. The connection is made at this level because we aim to build the structure of the kernel independently of any GUI, making it more flexible.

Figure 5 illustrates an overview of the implementation structure of the (except the packages in the *gui* layer which is out of this work).

7.2 Instantiation

Before evaluation, the needs of each docker must be first instantiated. We show below that of D_1.

The individual aspects

- $PN_{1/1}$: hunger⇒physiological (LN) *// when this need is in an insufficient state,* D_1 *can no longer move.*

- $PN_{1/2}$=f_1($PN_{1/1}$): transporting 3 ores⇒physiological (HN) *// the local need of the plan P. This HN is situated at level 1 because the docker knows that one of his basic motivations for doing the work is to get something to eat (via the money of the wage).*

- $PN_{1/3}$=f_2($PN_{1/1}$): going to the storage⇒ physiological (HN) *// the local need for reaching the storage location. It is situated at this level because it is the local need of a sub-action contributing to the realization of P (with* $PN_{1/4}$ *) at physiological level.*

$PN_{1/4}$=f_3($PN_{1/1}$): moving to the boat ⇒ physiological (HN) // *like* $PN_{1/3}$ *but concerning the boat location*

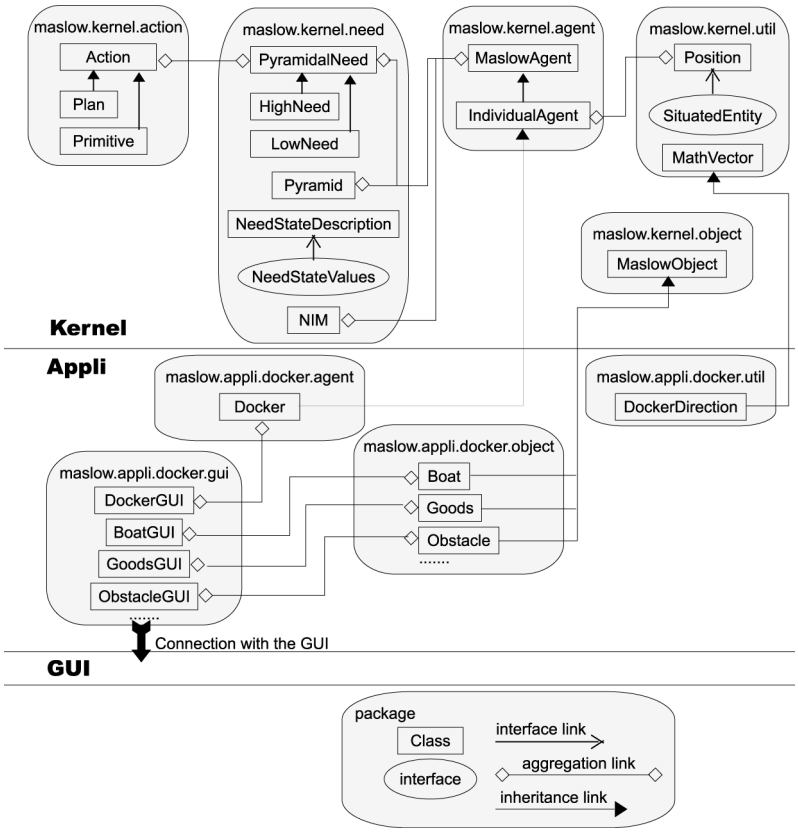

Fig. 5. A (partial) overview of the package structure of the implementation

---

- $PN_{2/1}$: safe from any grounded obstacles ⇒security (LN) // *avoiding any obstacles situated in the ground*
- $PN_{2/2}$: safe from any aerial obstacles⇒security (LN) // *avoiding any objects, e.g. that of falling from the crane.*
- $PN_{2/3}$: safe from any dangerous regions ⇒security (LN) // *regions are here the sea and the road-limit.*

The social aspect—
- $PN_{3/1}$: not to be alone⇒social (LN) // *just knowing or seeing that there is a congener around him (intrinsic characteristic of social entities)*
- $PN_{3/2}$: to be integrated or loved⇒social (LN) // *being accepted in a group. Remind that the group may also just be a set of animals, not always a cognitive structure having a common goal. We are here at a LN level.*
- $PN_{3/3}$: consideration⇒social (LN) // *respecting the needs in the pyramid of the other*
- $PN_{3/4}$:=clone($PN_{1/2}$)=f_4($PN_{3/2}$): transporting 3 ores⇒ social (HN) // *the local need of the plan P. This HN is a clone for this level 3 because the docker knows that one of his basic social motivations in doing the work perfectly is not to being rejected (it is a functional motivation as mentioned in Section 3.4).*
- $PN_{3/5}$=f_5($PN_{3/2}$): going to the storage⇒social (HN) // *the local need for reaching the storage location. It is situated at this level because it is the local need of a sub-action contributing to the realization of P (with $PN_{3/4}$) at social level.*
- $PN_{3/6}$=f_6($PN_{3/2}$). moving to the boat ⇒ social (HN) // *like $PN_{3/5}$ but concerning the boat location*
- $PN_{3/7}$=f_7($PN_{3/3}$): helping the others⇒social (HN) // *helping the other during the work (if possible)*

- $PN_{4/1}$: to be appreciated ⇒esteem (LN)
- $PN_{4/2}$:=clone($PN_{3/7}$)=f_8($PN_{4/1}$): helping the others⇒esteem (HN) // *helping the other during the work (if possible). The motivation is to be considered as a "good person" by D_2*

7.3 Scenario results

According to the above scenario, and regardless of D_2, we present here some behavior of D_1, when the two dockers are carrying goods:

- D_1 avoids D_2 when they meet each other. It is due not only to their respective preservation instinct (when they meet, $PN_{2/1}$.isLimit()) but also due to the consideration of the state of the needs of the other ($PN_{3/2}$.isLimit())
- likewise, the robots also avoid other obstacles (also due to $PN_{2/1}$). When an unexpected obstacle arrives close to an agent (for instance fallen down from the crane), $PN_{2/1}$.isInsufficient() is true. Then, the current plan P (motivated by $PN_{1/1}$) is immediately suspended (because of the criterion in Equation (4)) and an action among that corresponding to $PN_{2/1}$ is reactively chosen.

The behavior is reactive because $PN_{2/1}$ is at the insufficient state.
After this back tracking, the plan P is resumed to fulfill $PN_{1/2}$.

- in this scenario, it is noted that when D_1 has finished his own job (he has carried his 3 goods) while D_2 has not, D_1 will help D_2 in his work, because he knows that by doing so, he will improve the state of one of the needs of D_2 (satisfaction of $PN_{3/7}$, is motivated by $PN_{3/3}$).
- each time the water level of D_1 decreases, he drinks, leading this level to a better state again (illustrated in Figure Fig. 6).
- ...

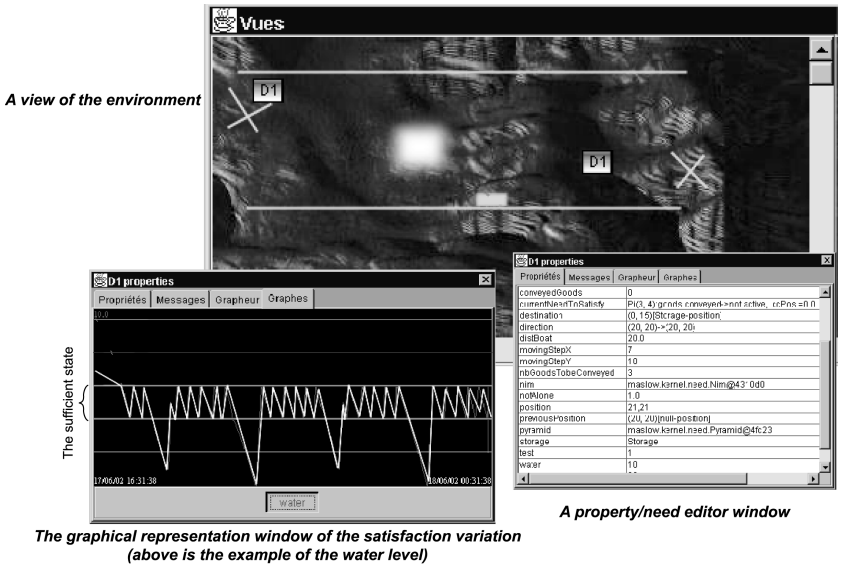

Fig. 6. The interface user of our simulation. In the scheme, we can see the variation of the need to drink of the Docker D1

8 Synthetical analysis of the model

8.1 Architecture and behavior

The hybrid model we propose here is a hierarchical model. But unlike many hybrid agent architectures also constructed from hierarchical functional layers, each level is composed of motivations. In fact, the notion of needs and motivations was not particularly well developed in hybrid agent theory (see Section 1.2 about the discussion of the state of art). Our present approach is an attempt to overcome this situation by modeling the source of the behavior rather than the behavior itself, as the latter is actually only the consequence of the former.

Likewise, the social concept in our model also concerns the social motivations of agents instead of high-level social concept such as cooperation [15]. We can say that our model complements them at a basic level.

Furthermore, in many hybrid models [13], [19], the reactive part has a semantic relation with the current mental state as well as the current planning of the agent. The reaction is then dependent on the current cognitive actions. In our model, this may or may not happen, i.e. that there is or is not semantic relation between cognitive and reactive behavior. An example of the first case is that of the docker D_1 who reactively takes another direction during the transport because his usual way contains unexpected obstacles (here, the semantic relation exists between the two levels). But in the second case where for instance D_1 drinks water during his work, this semantic relation does not exist. Thus, our low-level approach is more flexible. This flexibility is possible because the LN-needs in the pyramid of the agent do not take into account the current external context of the agent. What these needs "want" is to be satisfied, independently of the way this is performed (by current plan or not).

In addition, each level in the MASLOW hybrid architecture presents both reactive and cognitive parts of the agent behavior while generally, one level exclusively contains one of them [8], [15].

Finally, the LN concept in MASLOW is an abstract concept of needs such as hunger, thirsty, etc., found in many works about reactive agents [4], [9] also based on the determination of the degree of motivations. Actually, these needs are occurrences of LN at the appropriate levels.

8.2 Comparison to a previous work

Compared to [3], our more recent work, we have seen many evolutions in this paper:

- the list of criteria used for managing the need importance is added and some adjustments are fixed (see details in Section 6.2).
- the possible state of the need is also updated by adding the excessive state (Section 3.2). By doing so, our model is closer to the real-world situation
- conceptually, our previous work does not greatly develop the discussion of the hybrid agent paradigm while in this paper, it does. It only outlined the hybridism notion and the work was rather focused on the need importance as well as the action selection mechanism.

9 Conclusion

9.1 Summary

We present in this paper a model of a hybrid agent based on basic animal/human needs whose satisfaction constitutes the basic motivations of agent behavior in individual and social environment. We adopt this approach unlike many current hybrid models that are focused on the study of agent behavior or agent plan in

the composition of their hierarchical structure. Moreover, the pyramid we use for this work is a result of a real world investigation of the psychologist Abraham Maslow and as such, each level has a specific conceptual semantic not based on pure hypothesis. We treat both individual and social aspects of these needs.

9.2 Future work

Despite this ongoing research, much functionality will still be investigated in the near future.

Modeling the cognitive part The formalization of agent knowledge will be improved. Indeed, in multi-agent paradigm, the information that an agent has about its environment is generally partial [7] unlike our assumption in this paper that an agent has knowledge of the (whole) need state of the pyramid of others. For instance, it may happen that for one or another reason, agents intentionally hide their real need states. Consequently, there is no real certitude in some information about them, there is just an assumption or a belief. Thus, the concepts of *belief* and *intention* (to hide some states) should be considered. For this purpose, works such as [10], [21] seem to be interesting references for us.

By studying this cognitive part, we can extend the criteria in the determination of the need importance discussed in Section 6. Indeed, we also consider criteria determined at a purely cognitive level, for instance resulting from an individual or collective organization, roles, negotiation, and so on, between cognitive agents. These factors explicitly state that a given need PN_x must be considered first before a need PN_y, and according to given rules.

Miscellaneous Additional work will also be considered in the improvement of the model:

- integrating the general state of agents: we currently model the state of a given PN of an agent but not the general state of the pyramid, according to the set of all needs in it. This generalization is important because it allows the user to determine the global state of the agent itself
- improving the social modeling: in a real situation, the help action mentioned in Section 4.1 is not actually systematic. Generally, additional parameters must be taken into account before helping (the degree of relationship between the two dockers, individual objective of each docker, etc.). Moreover, such satisfaction of others' needs must also take into account the state of its own need before the decision.
- extending the notion of love as presented in Section 4.1: being loved by other agents such as family, friends, etc.

Acknowledgement

This research is funded by the University of the Indian Ocean. We thank all people who are working there. We are also thankful to everybody who improved the quality of this paper.

References

1. Andriamasinoro F., Courdier R. (2000) *A model of virtual competition between remote agents*, In proceedings of the 1^{st} Congress on Multi-Agent and Mobile-Agent (MAMA), Wollongong, Australia, december 12^{th} - 15^{th}
2. Andriamasinoro F., Courdier R. (2001) *Un Modèle Dynamique de Comportement Agents à base de Besoins*. In Amal El-Fallah Segrouchni, Laurent Magnin (eds.) : Journées Francophones sur l'Intelligence Artificielle Distribuée et les Systèmes Multi-Agents (JFIADSMA'01). Hermès Editions. November 12-15, Montreal, Canada.
3. Andriamasinoro F., Courdier R. (2002) *The Basic Instinct of Autonomous Cognitive Agents*. International Conference on Autonomous Intelligent System (ICAIS'2002), February 12th-15th, Geelong, Australia
4. Bonabeau E., Theraulaz G. (1994). *L'Intelligence Collective*. Edition Hermès.
5. Brazier F., Dunin-Keplicz B., Treur J., Verbrugge R. (1999). *Beliefs, Intentions and DESIRE. Modeling internal Dynamic behavior of BDI agents.* In JJ Meyer, PY Schobbens (eds.) Formal Models of Agents: ESPRIT project Modelage final workshop, Lecture Notes in AI, volume 1760, Springer, pp 36-56.
6. Daco P. (1965): *Les triomphes de la psychanalyse*, Marabout Service, MS 29
7. Ferber J. (1997). *Les systèmes multi-agents, vers une intelligence collective*. Inter-Editions.
8. Ferguson, I. A. (1992). TouringMachines: *An Architecture for Dynamic, Rational, Mobile Agents*. PhD thesis, Clare Hall, University of Cambridge, UK.
9. Gershenson Carlos, González Pedro Pablo, Negrete Jose Martinez (2000). *Action Selection Properties in a Software Simulated Agent.*, in Cairó et. al. (Eds.) MICAI 2000: Advances in Artificial Intelligence. Lecture Notes in Artificial Intelligence 1793, pp. 634-648. Springer-Verlag.
10. Grosz B. J, Kraus S. (1996). *Collaborative plans for complex group actions* in Artificial Intelligence 86, pp.269-357.
11. Jennings N. R (1996) *Coordination Techniques for Distributed Artificial Intelligence*, in Foundations of Distributed Artificial Intelligence (eds. G. M. P. O'Hare and N. R. Jennings), Wiley.
12. Jennings N. R.: Coordination TecHNiques for Distributed Artificial Intelligence, In G. M. P. O'Hare and N. R. Jennings (eds.): Foundations of Distributed Artificial Intelligence, Wiley, (1996).
13. Kurihara S., Aoyagi S., Onai R.(1997). *Adaptive Selection of Reactive/Deliberate Planning for the Dynamic Environment, - A Proposal and Evaluation of MRR-planning* -. In proceedings of the 8^{th} European Workshop on Modelling Autonomous Agents in a Multi-Agent World, (MAAMAW'97) Ronneby, Sweden.
14. Au S., Liang J., Parameswaran N (1998). *Progressive plan execution in a dynamic world* in Dynamic and Uncertain Environments Workshop in Artificial Intelligent Planning, Eds. *Ralph Bergmann, Alexdrader Kott,* Pittsburgh USA, AAAI.

15. Müller, J.P. (1994). *A conceptual model of agent interaction*. In Deen, S. M., editor, Draft proceedings of the Second International Working Conference on Cooperating Knowledge Based Systems (CKBS-94), pages 389-404, DAKE Centre, University of Keele, UK.
16. Maes P. (1991). *A Bottom-Up Mechanism for Action Selection in an Artificial Creature*. From Animals to Animats: Proceedings of the Adaptive Behavior Conference '91, edited by S. Wilson and J. Arcady-Meyer, MIT Press, February.
17. Malec J. (2000) *On Augmenting Reactivity with Deliberation in a Controlled Manner*. Workshop on Balancing Reactivity and Social Deliberation in Multi-Agent Systems at the 14th European Conference on Artificial Intelligence (ECAI), Berlin, Germany.
18. Maslow A. H. (2000). *The Maslow business Reader.*, edited by Collins D. S, J.Wiley & Sons, Inc.
19. Mavromichalis V. K., Vouros G. (2000). *ICAGENT: Balancing between Reactivity and Deliberation*. In Workshop on Balancing Reactivity and Social Deliberation in Multi-Agent Systems at the 14th European Conference on Artificial Intelligence (ECAI), Berlin, Germany.
20. Panzarasa P., Norman T., Jennings N. R. (1999) *Modeling sociality in the BDI framework* in Proceedings in 1^{st} Asia Pacific Conference on Intelligent Agent Technology, Hong-Kong.
21. Sullivan D., Grosz B., Kraus S. (2000): *Intention Reconciliation by Collaborative Agents*. 4^{th} International Conference on Multi-Agent Systems (ICMAS 2000), Boston USA, IEEE Computer Society Press.
22. Tambe M. (1996) *Executing Team Plans in Dynamic Multi-agent environments*. AAAI Fall Symposium on Plan Execution. Boston USA.
23. Wooldridge Michael (2002). *An Introduction To Multiagent* System, Published by John Wiley & Sons, March.