Towards Motivation-Based Decisions for Worth Goals

Steve J Munroe[1], Michael Luck[1], Mark d'Inverno[2]

[1] Electronics and Computer Science University of Southampton, Southampton, UK
{sjm01r, mml}@ecs.soton.ac.uk
[2] Computer Science, University of Westminster, London, UK
dinverm@wmin.ac.uk

Abstract. In this paper we present a motivational mechanism to generate and determine the worth of goals and to represent various constraints involved in satisfying a goal. The work builds on the SMART agent framework and adds to the growing body of work that is attempting to extend the abilities of autonomous agents past the constraints of the traditional symbolic approaches to AI. The paper represents a first step in increasing an agent's autonomy in the domain of e-commerce, specifically enabling the agent to dynamically set issue parameters in relation to the importance of the issue and the effects of any existing constraints.

1 Introduction

Overcoming the limitations of symbolically based representations as used in intelligent agents, to cope with more realistic domains, is an area growing in size. Work from robotic control [14], design to criteria scheduling [19] and cognitive appraisal theory [16] all pertain to extending the abilities of computational agents into continuous, worth-oriented domains. Trading-off the satisfaction of multiple issues or goals in the context of conflicting constraints however, has had little attention as yet in the field of autonomous deliberative agents (cf. [17]), though work relevant to this exists in auctions in the case of multiple-issues (eg. [2]) and robotic control in the case of synthesising behaviour in the face of multiple constraints (eg. [14]). Traditional deliberative agent architectures (eg. [11]) tend to employ a static representation of preferences that an agent must try to satisfy with little or no room for adjusting them in the light of new information. Often it is not possible however, to fully satisfy existing goals in a particular environment and, lacking any way to relax goals, they are therefore likely to be dropped, and the associated utility gain lost.

Within the field of electronic commerce, recent advances in agents are allowing automation of many activities usually performed by humans. Agents are now able to use simple negotiation strategies in order to purchase a good or a service under conditions that best satisfy a user's preferences (eg. [8]). The dominant approach to the worth-oriented nature of such a domain is to use utility-based, selfish maximising agents (eg, [5]). However, take up of these new systems is generally limited to simple purchases over one *issue* such as price, in which user preferences are rigidly encoded, offering little opportunity for flexible yet robust adaptation to prevailing circumstances. It would be desirable to be able to combine both approaches to see how to increase an agent's autonomy with respect to making purchasing decisions in the face of dynamic environments and changing contexts.

Consider the task of keeping an inventory stocked with various items; the inventory keeper needs to decide *how much* of each good to buy whilst trying to satisfy certain constraints such as cost, urgency and demand for each item. All of these constraints should be factors in the final decision and the inventory keeper needs to discover the feasible goals available given the constraints. The problems in this situation consist of how to use information about the domain in order to dynamically set parameters associated with the preferences inside the agent, how to discover when one issue under consideration is more important than another and, finally, how to combine the varying importance of each issue under consideration into a coherent decision about what to do. There are many kinds of domains in which this may be done but, in this paper we focus on domains in which the agent needs to reason over *quantities* of a particular item though this can easily be reinterpreted as reasoning over levels of *quality* for say, service provision.

In this paper our approach uses the concept of motivation to combine the traditional notion of goals as state descriptions with the notion of the worth or utility of a goal. This allows us to give a different worth to a goal state depending on the current context and constraints. We describe a motivational mechanism capable of generating multiple goals, representing their worth in relation to each other and determining their parameters based on a number of constraints (also modelled as motivation). We begin by describing the basic underlying framework upon which this work is based, before describing the motivational mechanisms in detail. Then we begin a discussion on assigning worth to goals and constraints, its relation to motivation and the setting of goal parameters. After that we provide a worked example. The paper ends with a discussion of the approach.

2 Motivated Agents

In this work we adopt the SMART agent framework described in [4]. We also adopt the Z notation [15] which is based on set-theory and first order logic. Though we assume some familiarity with Z, the meaning should be clear. The arguments for Z are well-rehearsed (ie.[3]), and we omit further discussion.

We start by defining four primitives: *attributes*, *actions*, *goals* and *motivations*, which are used as the basis for development of the SMART agent framework. Formally, these primitives are specified as given sets, which means that we say nothing about how they might be represented for any particular system. Attributes are simply features of the world, and are the only characteristics that are manifest. Actions are operations that can add or remove attributes, goals are descriptions of states in terms of attributes that the agent would like to bring about and, finally, motivation is any desire or preference that can lead to the generation and adoption of goals and that affects the outcome of the reasoning or behavioural task intended to satisfy those goals [4].

$$[Attribute, Action, Goal, Motivation]$$

An agent is defined to be an entity with a set of attributes that describes its characteristics, a set of actions called *capabilities* that can be used to change the world and a set of goals that the agent wants to bring about, all of which are non-empty sets.

$$
\begin{array}{|l}
\hline
Agent \\
attributes : \mathbb{P}\, Attribute;\ capabilities : \mathbb{P}\, Action \\
goals : \mathbb{P}\, Goal;\ motivations : \mathbb{P}\, Motivation \\
\hline
attributes \neq \varnothing \land capabilities \neq \varnothing \land goals \neq \varnothing \\
\hline
\end{array}
$$

An autonomous agent is an agent with the additional constraint of a non-empty set of motivations.

$$Autonomousagent = [Agent \mid motivations \neq \varnothing]$$

Some of an autonomous agent's capabilities are perceptual capabilities by which attributes in the environment can be captured in the agent's *view* of the world which is defined as a set of attributes.

$$View == \mathbb{P}\, Attribute$$

We present the agent's perceptual capabilities in the schema, $AutonomousAgentPerception$ below.

$$
\begin{array}{|l}
\hline
AutonomousAgentPerception \\
View : \mathbb{P}\, Attribute \\
\hline
\end{array}
$$

The kinds of domains we are interested in are *worth-oriented domains* [13], where it is possible to assign a *worth value* to all possible states in relation to a goal. A *worth goal* is defined in a similar fashion to traditional AI state goals but we need to develop methods that can extract the extra information needed to reason about such domains. We thus define a worth goal ($WGoal$) to be the same as a state goal, i.e. in terms of state.

$$WGoal == State$$

3 Motivation

The notion of motivation is increasingly being used as the basis for control of autonomous agents. Indeed, motivation has already been investigated in terms of goal generation [4], proactive behaviour [10] and information processing [9]. Perhaps the predominant view is that of motivation as a means to enable agents to *generate* goals within, rather than *adopt* them from other agents [4]. Goals are generated from motivations, higher-level non-derivative components that characterise the nature of an agent. They can also be considered as the desires or preferences that affect the outcome of a given reasoning or behavioural task. In a computational context, we can imagine a robot that normally explores its environment in an effort to construct a map, but must sometimes recharge its batteries. These motivations of 'curiosity' and 'hunger' lead to the generation of specific goals at different times, with a changing balance of importance as time passes.

Like the traditional notion of utility, motivation places value on actions and world states, but is a more wide-ranging concept than utility. In the traditional view, an agent

examines its options and chooses one with the highest utility. Whilst also performing this task, motivation is involved more intimately with the agent's decision-making process. A motivated agent has a dynamically changing internal environment, provided by motivations, that can influence its decisions. For example, in the presence of food, an agent may or may not choose to eat depending on the state of its internal environment (specifically its hunger motivation). In many systems, the utility for a given action or state is calculated in advance by the designer, but motivated agents can calculate utility on-the-fly based on weightings provided by their current motivational state.

We argue that the motivational approach offers a way to design more flexible agents that are able to act responsively within their sphere of concerns. More specifically, motivation offers a quantitative way for an agent to order a set of resource-conflicting goals in terms of importance within a given context. The actual ordering is therefore context-specific and changes with different sets of circumstances.

3.1 Intensity and Cues

In order to develop a computational model for motivation that can be manipulated and used in the way suggested above, we adopt the foundational work of d'Inverno and Luck [4] and Griffiths [6]. Much of what follows is based on this.

Motivation can be considered to be a dynamic control process [1], and the influence of a given motivation over an agent's decision-making can increase or decrease in response to changes in the environment or changes in the state of other motivations. We use the notion of *intensity* to capture this dynamic property. Intensity is here represented as a real valued number in the range [0,1] where 0 represents no intensity and 1 represents maximum intensity. The more intense a motivation, the more influence that motivation will exert over the decision-making process.

Thus, at any given time, each of an agent's motivations is characterised by an intensity that provides some indication of the motivation's appropriateness in the current environment. That is to say that when a motivation has high intensity, any goals generated should be highly *relevant* for the agent in the current environment. In order to achieve this, however, there must be some way of assessing the current environment in terms of relevance to motivations. The simplest way to achieve this is by attaching a set of *cues* to a motivation that determine when, and by how much, the motivation's intensity should be updated. These cues are similar to the *invocation conditions* of plans in BDI architectures [12].

In this way, a cue represents a *condition* which, when satisfied, results in a motivation's intensity being updated by some amount. This amount can either be fixed, or depend on some *measurement* of the environment. Each of these update methods calls for a different type of cue, called respectively *discrete* and *continuous*. In some cases, all an agent needs to know about a given world state in order to update its motivations is whether or not some proposition about the world state is true or false. *Discrete* cues provide this information to the agent. For example, a particular motivation may update its associated intensity by a fixed amount. Discrete cues represent sets of attributes and, as such, we can define them to contain a referent of type *attribute*:

$$DiscreteCue == \mathbb{P}_1\ Attribute$$

In the case of continuous cues an agent alters the intensity of its motivations in proportion to some measurement it takes from the environment. For example, an inventory

agent may update its motivation to keep the inventory well stocked by an amount proportional to the amount by which a stock item has been depleted. Here, a continuous cue might return the current amount remaining associated with the cue referent. We define continuous cues similarly to discrete cues, but consider further effects later on.

$$ContinuousCue == \mathbb{P}_1\ Attribute$$

Thus, the set of cues attached to a motivation can be a combination of both types of cues and as such we define cues to be either discrete or continuous cues.

$$Cue ::= dcue\langle\langle DiscreteCue\rangle\rangle \mid ccue\langle\langle ContinuousCue\rangle\rangle$$

Motivation is considered to have seven basic components: a unique identifier; a current and maximum intensity value; a set of goals that can be used to mitigate the motivation; a set of cues that lead to updating motivational intensity; and a discrete effect function and a continuous effect function. In order to define motivation we also need to define $[RAT_0^1]$ as the rationals between 0 and 1, assuming basic arithmetic operations are applicable [18]. The schema below provides a formal definition.

WMotivation

$$\begin{array}{l} name : MotiveSym;\ currentintensity, maxintensity : RAT_0^1 \\ goals : \mathbb{P}\ Goal;\ cues : \mathbb{P}\ Cue \\ discrete\_effect : Cue \rightarrow RAT_0^1 \\ continuous\_effect : Cue \rightarrow View \rightarrow RAT_0^1 \\ \hline \text{dom}\ discrete\_effect \subseteq \text{ran}\ dcue\ \text{dom}\ continuous\_effect \subseteq \\ \quad \text{ran}\ ccue\ maxintensity \leq 1\ currentintensity \leq maxintensity \end{array}$$

3.2 Updating Motivations

In order to update a motivation it is necessary to define update functions for both types of cue introduced above. Discrete cues are used in an update method called *discrete update* similar to Griffiths in [6], while continuous cues return update values that depend on measurements of the environment recorded through an agent's sensory processes; we refer to this update method as *measured update*. We explain each in turn below.

An agent's perceptual system forms a view of the environment that is composed of attributes. These attributes are checked against the cues attached to the agent's motivations to see if any cues are satisfied and, if so, the associated motivation is placed in the set of active motivations. To represent this formally, we define the function *selectActive*, which takes a state and set of motivations, and returns a set of motivations whose cues are satisfied.

$$\begin{array}{l} selectActive : View \rightarrow \mathbb{P}\ WMotivation \rightarrow \mathbb{P}\ WMotivation \\ \hline \forall\ v : View;\ ms : \mathbb{P}\ WMotivation \bullet selectActive\ v\ ms = \\ \quad \{m : ms \mid (\exists\ cs : Cue \bullet cs \in m.cues \wedge \\ \quad\quad (\bigcup \{m : m.cues \bullet (dcue \cup ccue)^{-1} m\} \subseteq v)) \bullet m\} \end{array}$$

We present discrete update as an axiomatic definition $dUpdate$. It takes a discrete cue of a motivation and increases its intensity to the minimum of either the maximum

allowed intensity or the updated intensity of the motivation as determined by the value of its cue.

$$dUpdate : Cue \rightarrow WMotivation \rightarrow WMotivation$$

$$\begin{aligned}
&\forall\, m : WMotivation;\ c : Cue \mid c \in (\text{ran}\ dcue) \land c \in m.cues \bullet \\
&\quad dUpdate\ c\ m = (\mu\, new : WMotivation \mid new.currentintensity = \\
&\quad \min\ (m.maxintensity, m.currentintensity + m.discrete_effect\ c) \land \\
&\quad new.goals = m.goals \land new.discrete_effect = m.discrete_effect \land \\
&\quad new.continuous_effect = m.continuous_effect \land new.cues = m.cues \land \\
&\quad new.name = m.name)
\end{aligned}$$

We define the continuous update function similarly, though we leave out the predicate as it is similar to that for discrete update.

$$cUpdate : Cue \rightarrow View \rightarrow WMotivation \rightarrow WMotivation$$

Next, we state what happens when a discrete cue is processed. A cue is input and the agent's perception is changed indicated by the delta prefix to the $AutonomousAgentPerception$ schema, the motivation associated with the cue is put into the class of active motivations using the $selectActive$ function and is then updated using $dUpdate$ function.

ProcessDiscreteCue

$$\begin{aligned}
&cue? : Cue \\
&\Delta AutonomousAgentPerception \\
&wmotivations, wmotivations' : \mathbb{P}\ WMotivation \\
&activatedmot, newmot : \mathbb{P}\ WMotivation \\
&\hline
&cue? \in (\text{ran}\ dcue) \\
&activatedmot = selectActive\ (dcue^{-1}\, cue?) \\
&wmotivations\ newmot = \{m : activatedmot \bullet dUpdate\ cue?\ m\} \\
&wmotivations' = wmotivations \setminus activatedmot \cup newmot
\end{aligned}$$

Continuous cues are treated similarly but with a modified third predicate. Here the motivation is updated using the $cUpdate$ function.

$wmotivations\ newmot = \{m : activatedmot \bullet cUpdate\ cue?\ view\ m\}$

4 Worth-Goals and Motivation

Now that we have a more detailed understanding and model of motivation, we can return to a consideration of *goal parameters*, and investigate how they are related to motivations. In our work we use the concept of *worth* in two ways. First, we propose a mechanism by which the worth of a goal is dynamically set as a function of the intensity of the underlying motivation. In this way, the worth of a goal provides a way of choosing between competing goals. Second, we determine the worth of any state in relation to a goal through the use of a *metric* by which we can measure the *proximity* of an environmental state to a goal. In this way, it is possible to make judgements about the *relative* satisfaction an environmental state offers in satisfying a goal.

For example, an agent that is highly motivated to eat (i.e. its hunger motivation has high intensity) would assign high worth to any generated goals which, when satisfied,

would mitigate its hunger. However, the very same goals would be assigned low worth if the hunger motivation of the agent was at low intensity. Thus the value or worth of a goal depends crucially on the underlying motivational intensity. Similarly, as the world state approximates the goal state (i.e of having eaten), it would receive a higher worth value. Below we show the signature of a function that calculates a goal's worth but omit details for space considerations.

$$Goalworth : WMotivation \rightarrow WGoal \rightarrow State \rightarrow RAT_0^1$$

In the above example however, the agent must also have a means to discover *how much* to eat. We refer to this as a *goal parameter*, and their values are also determined by the underlying motivation. Specifically, there must be some relation from the motivational intensity to the value placed on a goal parameter. For example, if an agent's hunger motivation was of high intensity then a parameter associated with a goal to eat could state that the agent eats a quantity close to the agent's eating capacity (we assume that parameters have a pre-defined range of acceptable values). Now, suppose that it is not possible to bring about the exact state defined by a goal, but only some approximation of it. How should one go about evaluating this state? The concept of a worth-oriented domain [13] allows for the development of a notion of *state-to-state proximity*, which can be represented by various metrics, depending on the scenario under consideration. For example, the proximity of the current state to the goal state above could be defined as the difference between the agent's current measure of satiation and the level defined in the goal's parameter.

By measuring the proximity of the current state to a goal state, we can assign a worth value to all states in relation to the goal. The effect of any given state on an underlying motivation is given by a *mitigation function*, which takes a goal, the associated motivation and the current world state defined as an agents current view of the environment, and returns a value that is used to *mitigate* the intensity of the underlying motivation. States that completely match those defined by a goal should maximally satisfy, or mitigate, the associated motivation. Below we show the signature of the mitigation function but again omit details due to space considerations.

$$mitigate : WGoal \rightarrow View \rightarrow Motivation \rightarrow RAT_0^1$$

Of course, to achieve this it must be possible to determine suitable metrics within the domain over which such characteristics as goal parameters, and state-to-state proximity can be defined. For example, in describing the inventory scenario we need to represent the sets of items that are used to stock the inventory, such as batteries, transistors, etc. One of the goals of an agent could specify the amount of one type of item to buy, say batteries = 500. This numerical value is a *goal parameter*, and represents the *ideal* parameter value, but this is likely to be adjusted in the face of currently active constraints. By constructing environments with metrics of suitable granularity, it becomes possible to calculate the proximity of one state to another and, consequently, to calculate the worth of an environmental state in relation to a goal state in terms of their proximity. Thus, as well as specifying a goal as a collection of attributes with certain relations, it is necessary to define certain goal parameters which represent *ideal values* for a particular attribute defined in the goal. For example, an ideal value could simply refer to the

quantity of some product, such as 300 transistors, or the *position* of some object, such as object a at location $20x,\ 30y$. Alternatively, an ideal value could be defined over a collection of attributes, such as the *amount paid* for both batteries and transistors.

5 A Worked Example

Here we describe in detail the inventory example introduced above. An agent must keep track of the number of items in the inventory and purchase appropriate amounts of each as they are consumed. For simplicity we consider only two types of stock items, transistors and batteries. The *maximum quantity* of transistors and batteries is 100 each. The agent also values each transistor at \$20 and each battery at \$10.

The agent also has a number of constraints. The amount of stock stored can never exceed the *maximum available space*. There is also a *maximum monthly budget* for purchasing stock, and a *current budget*, which must never exceed the maximum. Over time, stock will be consumed and the inventory will become depleted. Suppose the agent has three motivations, one for economy m_e (a constraint motivation), which represents a concern over money spent; one for time m_t (another constraint motivation), which is related to how full the inventory is (the more empty the inventory is the *more urgent* is the need to restock it) and finally one to ensure that the inventory is kept well stocked, m_r (a re-stock motivation).

Each motivation has appropriate cues: m_e has a continuous cue whose referent is the state of the purchasing budget, where *lower* budget values bring about *higher* intensity levels. Conversely, the intensity of m_t is *increased* the *more empty* the inventory becomes, while m_r *increases* in proportion to the *declining* levels of each stock type remaining in the inventory. At a specified time the agent checks each of the stock types in the inventory, and their current level of depletion changes the intensity of m_r. In this example, the depletion of each type of stock has a simple additive effect, but it is easy to imagine that different types of item could have different intensity effects depending on the relative need for those products.

For example, at the start of the month the agent checks its inventory and sees that transistors and batteries have dropped below their respective maximum amounts. As a result the agent updates its motivation to restock each of the stock items. The motivations for economy and urgency are also updated. First, the quantity of stock left in the inventory is used to update the intensity of m_r. Let us suppose that the effect of the stock depletion leaves m_r with an intensity value of 0.3, and that to bring the stocks back up to quota the agent must buy 10 transistors and 5 batteries (where the values represent the goal parameters). Next, the available budget (\$100) causes the m_e motivation to have an intensity of 0.2. Finally, m_t is updated by measuring the space left in the inventory. The store room has 10 m^2 of space left. The more space there is available the more urgent is the need to restock (as larger amounts take longer to process and install in the warehouse). Assume that the agent has only 10 hours in which to take delivery and install the required goods into the warehouse. Batteries take 1 hour per unit to process and transistors take 2 hours. Imagine that the intensity of m_t after calculating the space remaining is 0.5.

The cues determine the parameters of the agent's goals and the values taken on by the constraints. That is, exactly 10 batteries and 5 transistors are needed, but the total amount of time allocated for delivery of both batteries and transistors must not exceed

10 hours, and the total cost must not exceed $100. It may not be the case, however, that the ideal values attached to the goal parameters can be satisfied given the constraints. Indeed, if the agent were to re-stock on all the batteries and transistors it needed, the processing time would reach 20 hours and the total cost would be $250. Thus we need to discover the feasible goal parameter values given the constraints. However, we need to be certain to respect each motivation's relative importance. One way to proceed is first to order the concerns in terms of the intensity of their underlying motivations (which determines their importance), then satisfy the most important concern and then satisfy successively less important concerns, whilst at the same time not degrading the solution found in the preceding step. The time motivation has the highest intensity of 0.5 and thus its constraint is given the highest priority, followed by the re-stocking motivation, which has intensity of 0.3[3]. Finally, the economy motivation has the lowest intensity, 0.2, and its associated constraint is thus given the lowest priority. Mathematically, we can represent the problem as follows:

$$
\begin{aligned}
c_1 &: \quad x_1 + 2x_2 \leq 10 \\
g_1 &: \quad x_1 \quad\quad = 10 \\
g_2 &: \quad\quad\ x_2 = 5 \\
c_2 &: 20x_1 + 10x_2 \leq 100
\end{aligned}
$$

We can now take these values and pass them to a goal programming (GP) algorithm (a form of mathematical programming, see [7]) which will return the *best* values for each of the concerns, where *best* refers to values that first satisfy the requirements of the highest priority concern and then lower priority concerns. In this case, the GP algorithm gives us the values of transistors (x_1) and batteries (x_2) as both 3. This result satisfies constraints c_1 and c_2 whilst minimally deviating from goals g_1 and g_2. The GP returns the values that at best satisfy the ideal values of each concern, or at worst minimally deviates from those values.

6 Discussion

This work adds to the growing body of work (eg. [1], [9], [14], [19]) that is attempting to extend the abilities of autonomous agents past the constraints of traditional symbolic approaches characteristic of most work on intelligent agents. The paper represents a first step in increasing an agent's autonomy in the domain of e-commerce, specifically in determining both a goal's *worth* and the effects of constraints on the parameters of a goal.

A motivational mechanism was presented that enables an agent to reason about and make decisions concerning multiple concerns in a worth-oriented domain. It describes a technique for updating motivational intensity levels using a continuous measure of the environment. The mechanism also enables an agent to more flexibly respond to the prevailing context by varying the importance of a given goal in relation to other active goals. The mechanism also gives the agent the ability to discover optimal goal parameters such as ideal target value in the face of multiple constraints. The example presented

[3] As stated above, we do not specify any difference in importance between batteries and transistors, so we prioritise the attached goals arbitrarily, but such a priority is possible (and indeed desirable) to accommodate.

in this paper has been implemented and we have shown that an agent with this mechanism can effectively deal with a number of resource conflicting goals. Future work will involve extensive empirical experimentation and evaluation. Limitations of this work include the inability of the model to deal with agent-to-agent interaction and the simplistic domain example. As such, we intend to extend the motivational mechanism to include social motivations and to improve the model to cope with the dynamic on-line determination of a variety of negotiation parameters.

References

1. S. Allen. *Concern Processing in Autonomous Agents*, PhD Thesis, University of Birmingham, 2001.
2. David, E and Azoulay-Schwartz, R and Kraus S. Protocols and Strategies for Automated Multi-Attribute Auctions, *ICMAS-2002 Fourth International Conference on MultiAgent Systems*, 2002.
3. M. d'Inverno and M. Fisher and A. Lomuscio and M. Luck and M. de Rijke and M. Ryan and M. Wooldridge. Formalisms for Multi-Agent Systems. *KER*, 12:3, 1997.
4. M. d'Inverno and M. Luck. *Understanding Agent Systems*. Springer-Verlag, 2001.
5. P. Faratin and C. Sierra and N. R. Jennings. Negotiation Decision Functions for Autonomous Agents. *Journal of Robotics and Autonomous Systems*, 24:3-4, 159–182, 1998.
6. Griffiths, N. *Motivated Cooperation*. PhD Thesis, University of Warwick, 2000.
7. Ignizio, J.P. *Goal Programming and Extensions*. Lexington Books, 1976.
8. N. R. Jennings and M. Wooldridge. *Applications of Agent Technology*. N. R. Jennings and M. Wooldridge (eds.), 1998.
9. D. Moffat and N. Frijda. Where there's a will there's an agent. M. Wooldridge and N. R. Jennings (eds.) *Intelligent Agents: Theories, Architectures, and Languages*, LNAI Volume 890, 245–260, 1995.
10. T. J. Norman and D. Long Goal creation in motivated agents. M. Wooldridge and N. R. Jennings (eds.) *Intelligent Agents: Theories, Architectures, and Languages* LNAI Volume 890, 277–290, 1995.
11. Anand S. Rao AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. W. van der Velde and J. W. Perram (eds.) *Agents Breaking Away* LNAI 1038, 42–55, 1996.
12. Anand S. Rao and Michael P. Georgeff. BDI Agents: from theory to practice. 312–319. *Proceedings of the First International Conference on Multi-Agent Systems* ICMAS'95, 1995.
13. Rosenschein, J.S. and Zlotkin, G. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers* MIT Press, 1994.
14. E. Spier and D. McFarland. A Finer-Grained Motivational Model of Behaviour Sequencing. *In From Animals to Animats 4: Proceedings of SAB96*, 1996.
15. Spivey, M. *The Z Notiation, 2nd ed.* Prentice Hall, Hemel Hempstead, 1992.
16. A. Staller and P. Petta. Towards a tractable appraisal-based architecture In Ca namero, D.; Numaoka, C.; and Petta, P., eds., Workshop: Grounding Emotions in Adaptive Systems, 56–61. SAB'98: *From Animals to Animats*, 1998.
17. Wagner T. *Toward Quantified, Organizationally Centered, Decision Making and Coordination*. PhD Thesis, University of Massac, 2000.
18. Valentine, S.H. Bowen, J.P., Nicholls, J.E. (eds.) Putting Numbers into the mathematical toolkit. Z User Workshop, London 1992, *Workshops in Computing*. Berlin, 1993.
19. T. Wagner and V. Lesser. Design-to-Criteria Scheduling: Real-Time Agent Control. *Proceedings of AAAI 2000 Spring Symposium on Real-Time Autonomous Systems*. 89-96, 2000.