In Multi-Agent Systems: Theories, Languages and Applications - Proceedings of the Fourth Australian Workshop on Distributed Artificial Intelligence, C. Zhang and D. Lukose, (eds.), Lecture Notes in Artificial Intelligence, 1544, 58-73, Springer-Verlag, 1998

Motivated Behaviour for Goal Adoption

Michael Luck[*] Mark d'Inverno[‡]

[*] Department of Computer Science, University of Warwick, CV4 7AL, UK
mikeluck@dcs.warwick.ac.uk

[†] Cavendish School of Computer Science, Westminster University, London W1M 8JS, UK
dinverm@westminster.ac.uk

Abstract. Social behaviour arises as a result of individual agents cooperating with each other so as to exploit the resources available in a rich and dynamic multi-agent domain. If agents are to make use of others to help them in their tasks, such social behaviour is critical. Underlying this cooperation is the transfer or adoption of goals from one agent to another, a subtle and complex process that depends on the nature of the agents involved. In this paper we analyse this process by building upon a hierarchy previously constructed to define objects, agents and autonomous agents. We describe the motivated *self-generation* of goals that defines agent autonomy and the adoption of goals between agents that enables social behaviour. Then we consider three classes of goal adoption by objects, agents and autonomous agents. The first of these is merely a question of instantiation, the second requires an understanding of the relationship of the agent to others that are engaging it, and the third amounts to a question of negotiation or persuasion.

1 Introduction

The notion of *autonomy* has associated with it many variations of meaning. According to Steels, autonomous systems must be automatic systems and, in addition, they must have the capacity to form and adapt their behaviour while operating in the environment. Thus traditional AI systems and most robots are automatic but not autonomous — they are not independent of the control of their designers [29]. Autonomous systems are independent and exercise *self-control*. To do this, it is argued, they must be *motivated*.

Autonomous agents possess goals which are *generated* within rather than *adopted* from other agents. These goals are generated from *motivations* which are higher-level non-derivative components characterizing the nature of the agent. For example, consider the motivation *greed*. This is not a goal in the classical artificial intelligence sense since it does not specify a state of affairs to be achieved, nor is it describable in terms of the environment. However, it may, if other motivations permit, give rise to the generation of a goal to rob a bank. The distinction between the motivation of greed and the goal of robbing a bank is clear, with the former providing a reason to do the latter, and the latter specifying what must be done.

This view of autonomous agents is based on the generation and transfer of goals between entities. Specifically, an entity is an agent if it can be viewed as satisfying a goal. This goal must first be created and then, if necessary and appropriate, transferred to another entity. It is this adoption of goals that changes an entity from an object to an agent, and it is the *self-generation* of goals that is responsible for its autonomy.

Key to understanding the nature and behaviour both of individual agents, and of any interactions between them, is this notion of autonomy. In a series of papers, we have described and formally specified an extended theory of agent interaction, based on *goals* and *motivations*, which takes exactly this standpoint. The theory describes a framework for categorising different agents [14], which has been used as a basis for investigating aspects of the relationships between agents [16], providing an operational account of their invocation and destruction [6] and analysing their complexity [8], as well as for reformulating existing systems and theories [3,4,7]. In all this, however, one aspect has either been omitted or only briefly alluded to, namely a detailed account of the generation of goals from motivations, and goal adoption between agents. This paper addresses that omission, by showing how the formal framework may be used to provide a detailed operational account of the processes of goal generation and adoption.

First, we provide some context for the concept of motivation and its use in directing reasoning and behaviour with a short review of related work, and then consider the role of motivation in autonomous behaviour in more detail. Section 4 provides a brief outline of the formal agent framework, giving a selection of Z schemas that describe salient aspects, so that a reasonable context is available within which to situate this work. Then we analyse how motivations are used in goal generation and subsequently goal adoption. At each point we describe the processes involved both informally and formally using the Z notation. Finally, we summarise and and present concluding remarks.

2 What and Why Motivation?

According to Halliday, the word *motivation* does not refer to a specific set of readily identified processes [9]. It is frequently discussed in terms of *drive* and *incentive*. Drives are related to physiological states such as the deprivation of food, hormones, etc, while incentives refer to external stimuli that affect motivation such as the presence of food, as an incentive to eat. Research on motivation is currently being pursued from a variety of perspectives including psychology and ethology. Our focus, however, is on providing an effective control mechanism for governing the behaviour and reasoning of autonomous agents through the use of motivations. Though we focus on a computational approach, in this section we will discuss related work.

Some psychological research has recognised the role of motivations in reasoning in a similar way to that suggested here. Kunda [12] informally defines motivation to be, “any wish, desire, or preference that concerns the outcome of a given reasoning task” and suggests that motivation affects reasoning in a variety of ways including the accessing, constructing and evaluating of beliefs and evidence, and decision making. Such arguments are supported by a large body of experimental research, but no attempt is made to address the issue of how motivations may be represented or applied in a computational context.

Computational work has also recognised the role of motivations. Simon [25] takes motivation to be “that which controls attention at any given time,” and explores the relation of motivation to information-processing behaviour, but from a cognitive perspective. Sloman [27,26] has elaborated on Simon’s work, showing how motivations are relevant to emotions and the development of a computational theory of mind.

Problem solving can be considered to be the task of finding actions that achieve the current goals. Typically, goals are presented to systems without regard to the problem-solving agent so that the reasoning process is divorced from the reality of an agent in the world. Clearly, this is inadequate for research concentrating on modelling autonomous agents and creatures, which requires an understanding of how such goals are generated and selected. Additionally, it is inadequate for research that aims to provide flexibility of reasoning in a variety of contexts, regardless of concerns with modelling artificial agents. Such flexibility can be achieved through the use of motivations which can lead to different results even when goals remain the same [13].

In proposing to develop a 'computational architecture of a mind', Sloman makes explicit mention of the need for a "store of 'springs of action' (motives)" [27]. In the same paper, he tries to explicate his notion of a *motive* as being a representation used in deciding what to do, including desires, wishes, tastes, preferences and ideals. The key feature of a motive, according to Sloman, is not in the representation itself, but its role in processing. Importantly, Sloman distinguishes between motives on the one hand, and 'mere subgoals' on the other. "Sometimes," he claims, "a mere subgoal comes to be valued as an end," because of a loss of 'reason' information. First-order motives directly specify goals, while second-order motives generate new motives or resolve conflicts between competing motives — they are termed *motive generators* and *motive comparators*. "A motive produced by a motive generator may have the status of a desire." This relatively early work presents a broad picture of a two-tiered control of behaviour: motives occupy the top level, providing the *drive* or *urge* to produce the lower level goals that specify the behaviour itself. In subsequent work, the terminology changes to distinguish between *nonderivative* motivators or goals and *derivative* motivators or goals, rather than between motivators and goals themselves. Nevertheless, the notion of derivative and nonderivative mental attitudes makes one point clear: that there are two levels of attitude, one which is in some sense innate, and which gives rise to the other which is produced as a result of the first.

In a different context, the second of Waltz's 'Eight Principles for Building an Intelligent Robot' requires the inclusion of "innate *drive* and evaluation systems to provide the robot with moment-to-moment guidance for its actions."[30] In elaborating this principle, Waltz explains that the action of a robot at a particular time should not just be determined by the current sensory inputs, but also the "desires" of the robot, such as minimizing energy expenditure (laziness), and maintaining battery power levels (hunger). This research into robotics, artificial life, and autonomous agents and creatures has provided the impetus for a growth of interest in modelling motivations computationally, and a number of different representations for motivations and mechanisms for manipulating them have been developed at both subsymbolic and symbolic levels (eg. [1,10]).

3 Motivated Behaviour in Autonomous Agents

A given stimulus does not always evoke the same response. If the external situation is constant, differences in response must be ascribed to changes in the internal state of the responding agent. These differences are due to the motivations of the agent.

An agent possesses a fixed range of identifiable motivations of varying strength. These motivations can be regarded as being innate, and certain behaviours may be associated with one or more motivations. For example, the behaviour of feeding is associated with the motivation of obtaining food, or hunger. In most cases, the execution of such a behaviour reduces the strength of the associated motivations, so that in the case of feeding, the motivation to obtain food is reduced. These behaviours are known as *consumatory behaviours*; other behaviours which are not associated with any particular motivation, but which make the conditions of a consumatory behaviour come true are known as *appetitive behaviours*. For example, a go-to-food behaviour might make the conditions (that there is food within reach) of the feeding behaviour become true.

This view of motivation is somewhat simplified, and although much behaviour occurs in functional sequences with appetitive behaviours leading to consumatory ones, complex interactions between motivations and behaviours are possible [11]. For example, a single factor could directly cause many activities, or cause an action which in turn leads to other behaviours, or even cause some motivations to decrease so that others would increase in turn. In addition there are inhibitory relationships between behaviours in animals and also relationships that increase the strength of other behaviours. Moreover, the combination of motivations may lead to different or variable behaviours. These are all difficult issues which must be addressed in attempting to construct accurate behavioural models of real and artificial agents. Our concern, however, is not with providing such accuracy, but in constructing *simple* yet *adequate* models which will allow effective control of behaviour.

We can define autonomous agents to be agents with a higher-level control provided internally by motivations. Thus we can specify motivations of *curiosity*, *safety*, *fear*, *hunger*, and so on. In a simple agent design, we might then associate the motivation of *curiosity* with the goal of *avoiding obstacles* which, in turn, is associated with the actions required to achieve such results. Motivations will also vary over time according to the internal state of the agent. For example, if the agent spends a long time without food, then the hunger motivation will increase. When the agent feeds, the hunger motivation will decrease.

Each motivation thus has a strength associated with it, either variable depending on external and internal factors, or fixed at some constant value. A motivation can thus be represented by a triple, $< m, v, b >$ known as an *m-triple* where m is the kind of motivation, v is a real number, the strength (or intensity [26]) value associated with that motivation, and b is a boolean variable taking the value *True* when the strength value, v, is fixed, and *False* when it is variable. An autonomous agent can be regarded as embodying a set of n motivations, M, which comprises the *m-triples*, $< m_1, v, b > \ldots < m_n, v, b >$. Thus the set of motivations, M, is a function of the kind of agent being considered, while each motivation in this set at a particular point in time is a function of an instance of a particular kind of agent and its environment together. In order to act on motivations, a threshold value for strength may be necessary, which must be exceeded to force action. Alternatively, the highest strength value may be used to determine the motivation currently in control.

More sophisticated mechanisms are possible such as those described by Norman and Long [22,23], Sloman [2,26] and Moffat and Frijda [21,20]. In addition, other rep-

resentations for motivations and mechanisms for manipulating them have been developed at both subsymbolic and symbolic levels (eg. by Schnepf [24], Maes [17–19] and Halperin [10]). All are possible instantiations of the model described in the remainder of this paper, but the details are unimportant at present. It is enough to note that the abstract model provides the framework within which such mechanisms can be incorporated according to the particular need.

4 The Agent Framework

As has been described elsewhere in more detail [14], we propose a four-tiered hierarchy comprising *entities*, *objects*, *agents* and *autonomous agents*. The basic idea underlying this hierarchy is that all components of the world are entities. Of these entities, some are objects, of which some, in turn, are agents and of these, some are autonomous agents. In this section, we briefly outline the agent hierarchy. Many details are omitted — a more complete treatment can be found in [14].

Entities can be used to group together attributes for any useful purpose without adding a layer of *functionality* on top. They serve as a useful abstraction mechanism by which they are regarded as distinct from the remainder of the environment, and which can organise perception. An object is just something with abilities and attributes and has no further defining characteristics. An agent is just an object that is useful, typically to another agent, where this usefulness is defined in terms of satisfying that agent's goals. In other words, an agent is an object with an associated set of goals. One object may give rise to different instantiations of agents which are created in response to another agent. This definition of agency relies upon the existence of these other agents which provide goals that are adopted in order to instantiate an agent. In order to escape an infinite regress of goal adoption, we can define autonomous agents which are just agents that can generate their own goals from motivations.

For example, a table can be an object. It has attributes specifying that it is stable, made of wood, is brown in colour and has a flat surface. Its capabilities specify that it can support things. If I support my computer on a table, however, then the table is my agent for supporting the computer. The table may not actually possess the goal, but it is certainly satisfying, or can be *ascribed*, my goal to support the computer. A robot which rivets a panel onto a hull is also an agent, and if it has motivations such as hunger and achievement, then it is an autonomous agent.

Mathematically, we can describe this view of agents and provide a complete formal specification of it using the Z specification language. Below, we present the basic components of the framework. Our use of the notation should be self-explanatory and we will not, therefore, give excessive detail here, though details can be found in [28].

Before we can move to a definition of any of these entities, we must first define some primitives. First, *attributes* are simply features of the world, and are the only characteristics which are manifest. They need not be perceived by any particular entity, but must be potentially perceivable in an omniscient sense. Second, *actions* are discrete events which change the state of the environment. Third, *goals* are describable states of affairs to be achieved in the environment. Finally, *motivations* are any desires or

preferences that can lead to the generation and adoption of goals and which affect the outcome of the reasoning or behavioural task intended to satisfy those goals.

We define an entity to have a non-empty set of attributes, as just something identifiable. An object is then an entity with a non-empty set of actions or capabilities.

$$Entity == [attributes : \mathbb{P}\, Attribute;\ capableof : \mathbb{P}\, Action;\ goals : \mathbb{P}\, Goal;\\ motivations : \mathbb{P}\, Motivation;\ |\ attributes \neq \{\ \}]$$

Similarly, an agent is an object with a non-empty set of goals, and an autonomous agent is an agent with non-empty motivations. Note the use of schema inclusion for incremental definition, by which earlier schemas are included and used subsequently.

$$Object == [Entity \mid capableof \neq \{\ \}]$$
$$Agent == [Object \mid goals \neq \{\ \}]$$
$$AutonomousAgent == [Agent \mid motivations \neq \{\ \}]$$

In summary, if there are attributes and capabilities, but no goals, then the entity is an *object*. If there are goals but no motivations, then the entity is an *agent*. Finally, if neither the motivation nor goal sets are empty, then the entity is an *autonomous agent*. Thus, we have constructed a formal specification which identifies and characterises agents and autonomous agents. Most usefully, perhaps, the specification is constructed in such a way as to allow further levels of specification to be added to describe particular agent designs and architectures.

5 Goal Generation

The framework described above involves the generation of *goals* from *motivations* in an autonomous agent, and the adoption of goals by, and in order to create, other agents. In this section, we build on earlier initial work in outlining goal generation and adoption [5]. We give a complete description and specification of how an autonomous agent, *defined* in terms of its high-level and somewhat abstract *motivations*, can construct goals. Earlier work, while describing the introduction of goals, did not consider the later stages of releasing entities from agency obligations, an omission corrected here.

An autonomous agent will try to find a way to mitigate motivations, either by selecting an action to achieve an existing goal as above for simple agents, or by retrieving a goal from a repository of known goals. Thus, our model requires a repository of known *goals* which capture knowledge of limited and well-defined aspects of the world. These goals describe particular *states* or *sub-states* of the world with each autonomous agent having its own such repository.

As stated elsewhere [16], in order to retrieve goals to mitigate motivations, an autonomous agent must have some way of assessing the effects of competing or alternative goals. Clearly, the goals which make the greatest positive contribution to the motivations of the agent should be selected unless a greater motivational effect can be achieved by *destroying* some subset of its goals. The motivational effect of generating or destroying goals not only depends on the motivations but also on the goals of the agent. For example, an autonomous agent should not generate a goal that it already possesses or that is incompatible with the achievement or satisfaction of its existing goals.

Formally, the ability of autonomous agents to assess goals is given in the next schema, $AssessGoals$. The schema describes how an autonomous agent monitors its motivations for goal generation. First, the $AutonomousAgent$ schema is included and the new variable representing the repository of available known goals, $goalbase$ is declared. Then, the motivational effect on an autonomous agent of satisfying a set of new goals is given. The $motiveffectgenerate$ function returns a numeric value representing the motivational effect of satisfying a set of goals with a particular configuration of motivations and a set of existing goals. Similarly, the $motiveffectdestroy$ function returns a numeric value representing the motivational effect of removing some subset of its existing goals with the same configuration. The predicate part specifies that the goal base is non-empty, and that all the current goals must be goals that exist in the goalbase. For ease of expression, we also define a $satisfygenerate$ function, related to $motiveffectgenerate$, which returns the motivational effect of an autonomous agent satisfying an additional set of goals. The $satisfydestroy$ function is defined analogously.

$$
\begin{array}{|l}
\hline
AssessGoals \\
\hline
AutonomousAgent \\
goalbase : \mathbb{P}\, Goal \\
motiveffectgenerate : \mathbb{P}\, Motivation \rightarrow \mathbb{P}\, Goal \rightarrow \mathbb{P}\, Goal \rightarrow \mathbb{Z} \\
motiveffectdestroy : \mathbb{P}\, Motivation \rightarrow \mathbb{P}\, Goal \rightarrow \mathbb{P}\, Goal \rightarrow \mathbb{Z} \\
satisfygenerate : \mathbb{P}\, Goal \rightarrow \mathbb{Z} \\
satisfydestroy : \mathbb{P}\, Goal \rightarrow \mathbb{Z} \\
\hline
goalbase \neq \{\} \\
goals \subseteq goalbase \\
\forall\, gs : \mathbb{P}\, goalbase \bullet \\
\quad (satisfygenerate\ gs = motiveffectgenerate\ motivations\ goals\ gs) \land \\
\quad (satisfydestroy\ gs = motiveffectdestroy\ motivations\ goals\ gs) \\
\hline
\end{array}
$$

Now we can describe the generation of a new set of goals in the $GenerateGoals$ operation schema. First, the agent changes, indicated by $\Delta AutonomousAgent$, and the previous schema is included. The predicate part simply states that there is a set of goals in the goalbase that has a greater motivational effect than any other set of goals, and the current goals of the agent are updated to include the new goals.

$$
\begin{array}{|l}
\hline
GenerateGoals \\
\hline
\Delta AutonomousAgent \\
AssessGoals \\
\hline
goalbase \neq \{\ \} \\
\exists\, gs : \mathbb{P}\, Goal \mid gs \subseteq goalbase \bullet \\
\quad (\forall\, os : \mathbb{P}\, Goal \mid os \in (\mathbb{P}\, goalbase) \bullet \\
\qquad (satisfygenerate\ gs \geq satisfygenerate\ os) \land \\
\qquad\quad goals' = goals \cup gs) \\
\hline
\end{array}
$$

Once generated by an autonomous agent, goals exist in multi-agent system until, for whatever reason, they are explicitly destroyed by that autonomous agent. This represents the end of the life of a goal. The destruction of goals is defined in a similar

way to the generation of goals, and formally in $DestroyGoals$. This schema states that an agent destroys the subset of its goals, the destruction of which provide the greatest motivational benefit.

$$
\begin{array}{|l}
\hline
DestroyGoals \\
\Delta AutonomousAgent \\
AssessGoals \\
\hline
goalbase \neq \{\} \\
\exists\, gs : \mathbb{P}\, Goal \mid gs \subseteq goalbase \bullet \\
\quad (\forall\, os : \mathbb{P}\, Goal \mid os \in (\mathbb{P}\, goalbase) \bullet \\
\qquad (satisfydestroy\ gs \geq satisfydestroy\ os) \land \\
\qquad\quad goals' = goals \setminus gs) \\
\hline
\end{array}
$$

6 Goal Adoption

Since we are interested in multi-agent worlds, we must consider the world as a whole rather than just individual agents. In this world, all autonomous agents are agents and all agents are objects. We also identify further sub-categories of entity. Before proceeding, therefore, we distinguish those objects which are not agents, and those agents which are not autonomous and refer to them as *neutral-objects* and *server-agents* respectively.

An agent is then either a server-agent or an autonomous agent, and an object is either a neutral-object or an agent.

$$NeutralObject == [Object \mid goals = \{\}]$$
$$ServerAgent == [Agent \mid motivations = \{\}]$$

We can then describe the world as a collection of neutral objects, server agents and autonomous agents.

$$
\begin{array}{|l}
\hline
World \\
entities : \mathbb{P}\, Entity \\
objects : \mathbb{P}\, Object \\
agents : \mathbb{P}\, Agent \\
autonomousagents : \mathbb{P}\, AutonomousAgent \\
neutralobjects : \mathbb{P}\, NeutralObject \\
serveragents : \mathbb{P}\, ServerAgent \\
\hline
autonomousagents \subseteq agents \subseteq objects \\
agents = autonomousagents \cup serveragents \\
objects = neutralobjects \cup agents \\
\hline
\end{array}
$$

In multi-agent systems agents may wish, or need, to use the capabilities of other entities. They can make use of the capabilities of these others by *adopting* their goals. For example, if Anne needs to move a table which cannot be lifted alone, she must get someone else to adopt her goal before it can be moved. Similarly, if she wants

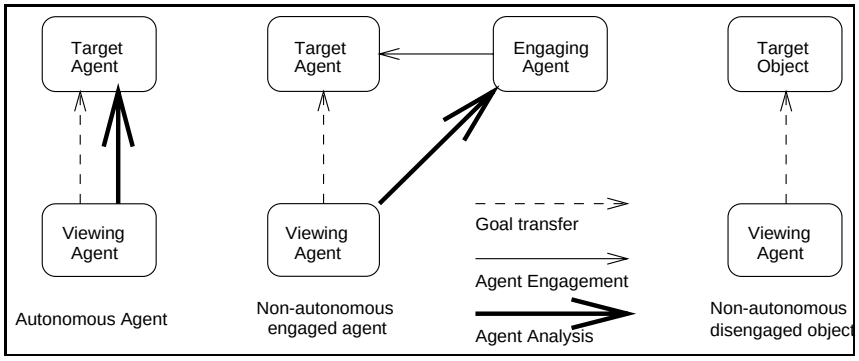

Fig. 1. Goal Adoption in Non-Autonomous and Autonomous Agents

tea, then she must make use of a kettle to boil water, a teapot to make the tea and subsequently a cup from which to drink it. Each of these objects can be ascribed, or viewed, as adopting Anne's goals in order that her thirst can be relieved. This notion of goal adoption underlies social behaviour, and an understanding of the ways in which in can be achieved is fundamental for effective modelling and simulation. In general, entities may serve the purposes of others by adopting their goals. However, the ways in which they adopt goals depends on the kind of object. They may be either neutral-objects, server-agents or autonomous agents, and each requires a separate analysis by the agent with a goal to be adopted, which we call the *viewing* agent.

In the description given in the previous section, goals may be generated only by autonomous agents. Both non-autonomous (server) and autonomous agents, however, can adopt goals. With autonomous agents, goal adoption amounts to a problem of *negotiation* or *persuasion*, requiring an analysis of the *target* autonomous agent. With non-autonomous agents, goal adoption requires an analysis of both the agent intended to adopt the goal, and any other agent *engaging* that agent. With objects, no analysis is required, since agents are *created* from objects with the relevant associated goals.

Figure 1 shows three fundamental cases of goal adoption which we consider in detail below. In the figure, there are three kinds of agent. A *target* agent or object is one that is intended to adopt goals. An *engaging* agent is one whose goals are currently (already) adopted by the target agent. A *viewing* agent is an agent that seeks to engage a target agent or object by having it adopt goals. It is a viewing agent because the way in which goal adoption is attempted is determined by its view of the situation. We consider the three cases of goal adoption below.

6.1 Goal Adoption by Neutral Objects

In the simplest case, goal adoption by non-autonomous agents occurs by instantiating an agent from a neutral object with the goals to be adopted. In this case, no *agent* exists before the goals are adopted, but the act of goal transfer causes an agent to be created from a neutral object using those particular goals. Thus, for example, a cup in Anne

and Bill's office, which is just an neutral-object, becomes an agent when it is used for storing Anne's tea. In this case it *adopts* or *is ascribed* her goal of storing liquid. It is possible to create the agent from the object because the cup is not being used by anyone else; it is not *engaged* by another agent. An entity can only be a neutral object if it is not *engaged*.

Below, we specify a function that creates a new entity by ascribing a set of goals to an existing entity. It is *total*, valid for any entity and any set of goals, and is just a formal way of associating goals with an entity, and will be used subsequently.

$$EntityAdoptGoals : (Entity \times \mathbb{P}\, Goal) \rightarrow Entity$$

$$\begin{aligned}
&\forall\, gs : \mathbb{P}\, Goal;\ old, new : Entity \bullet \\
&\quad EntityAdoptGoals(old, gs) = new \Leftrightarrow new.goals = old.goals \cup gs \\
&\qquad \land\ new.capableof = old.capableof \land new.attributes = old.attributes
\end{aligned}$$

We now specify how a non-autonomous disengaged object, or neutral-object is instantiated as a server-agent. In Z, a variable with a '?' indicates an input. Thus, a neutral-object and a set of goals are input, the entities in the world change, indicated by $\Delta World$, and the sets of objects and agents are updated accordingly. First, the set of neutral objects no longer includes the originally disengaged object. Second, the set of server agents now includes the newly created server-agent. Finally, the schema states that there is no change to the set of autonomous agents. In addition, the variables, $entities$, $objects$ and $agents$, are updated by removing the neutral-object and adding the newly instantiated server-agent. The final three predicates are redundant since they necessarily follow from the previous predicates, but are included to detail how all the state variables are affected. In subsequent schemas, such redundancy is not included.

NeutralObjectAdoptGoals

$$\begin{aligned}
&o? : NeutralObject \\
&gs? : \mathbb{P}\, Goal \\
&\Delta World \\
&\hline
&o? \in neutralobjects \\
&neutralobjects' = neutralobjects \setminus \{o?\} \\
&serveragents' = serveragents \cup \{EntityAdoptGoals\ (o?, gs?)\} \\
&autonomousagents' = autonomousagents \\
&entities' = entities \setminus \{o?\} \cup \{EntityAdoptGoals\ (o?, gs?)\} \\
&objects' = objects \setminus \{o?\} \cup \{EntityAdoptGoals\ (o?, gs?)\} \\
&agents' = agents \setminus \{o?\} \cup \{EntityAdoptGoals\ (o?, gs?)\}
\end{aligned}$$

For completeness, we specify the related operation where an entity is released from all of its agency obligations. Here, a server-agent reverts from a server-agent to a neutral-object. It is possible that a cup can be engaged for two separate goals. For example, it may be engaged as a vase for flowers and as a paper-weight for loose papers if there is a breeze coming from a nearby open window. If the window is closed and the flowers are removed, the cup is released from all its agency obligations and reverts to being a neutral-object.

Formally, this operation is defined by the $RevertToNeutralObject$ schema. It uses the axiomatic function, $EntityRemoveGoals$, defined similarly to $EntityAdoptGoals$, which removes a set of goals from an entity.

$$EntityRemoveGoals : (Entity \times \mathbb{P}\, Goal) \rightarrow Entity$$

$$\begin{aligned}
&\forall\, gs : \mathbb{P}\, Goal;\ old, new : Entity \bullet \\
&\quad EntityRemoveGoals(old, gs) = new \Leftrightarrow new.goals = old.goals \setminus gs \\
&\qquad \land\ new.capabilities = old.capabilities \land new.attributes = old.attributes
\end{aligned}$$

The predicates in the following schema check that the input goals are the same as the set of current goals of the server-agent. This ensures that the server-agent is released from *all* of its agency obligations. The variables, $neutralobjects$, $serveragents$ and $autonomousagents$, are updated accordingly.

$RevertToNeutralObject$

$$\begin{aligned}
&sa? : ServerAgent \\
&gs? : \mathbb{P}\, Goal \\
&\Delta MultiAgentSystem \\
&\hline
&sa? \in serveragents \\
&gs? = sa?.goals \\
&neutralobjects' = neutralobjects \cup \{EntityRemoveGoals\ (sa?, sa?.goals)\} \\
&serveragents' = serveragents \setminus \{sa?\} \\
&autonomousagents' = autonomousagents
\end{aligned}$$

6.2 Goal Adoption by Server Agents

If the target object is *engaged* by other agents then it is itself an agent, so the protocol for goal adoption changes. In this case, there are several ways to *engage* the target object.

The first involves supplying the target object with more goals that does not affect the existing agency obligations. In this case the agent is *shared* between the viewing agent and the existing engaging agents. The second involves trying to persuade any engaging agents to *release* the engaged object so that it becomes a *neutral*-object and can therefore subsequently be engaged by the viewing agent as required. (This may relate to the issue of goal adoption for autonomous agents, which is considered later). The third possibility involves *displacing* the engaging agent so that the engaged object becomes a neutral-object and can then subsequently be ascribed other goals. This possibility is dangerous since it may cause conflict with the previous engaging agents.

As an example, suppose that a cup is currently in use as a paper-weight for Anne, so that the cup is *Anne's* agent with her goal of securing loose papers. Suppose also, that Bill wishes to use the cup to have some tea. The first way for Bill to engage the cup is for him to attempt to use the cup without destroying the existing agency relationship between Anne and the cup. Since this would involve an awkward attempt at making tea in, and subsequently drinking from, a stationary cup, he may decide instead to try other alternatives. The second alternative is to negotiate with Anne to release the cup so that it can be used for storing tea while the third alternative is for Bill to displace the goal

ascribed to the cup by removing the cup from the desk and pouring tea into it. The cup is no longer an agent for Anne and is now ascribed the goal of storing tea for Bill. It has switched from being engaged by Anne to being engaged by Bill, and this is equivalent to the agent reverting to an object and then being re-instantiated as a new agent. This method may not be an appropriate strategy, however, because in destroying the agency obligation of the cup as a paper-weight, there is a risk of conflict between Anne and Bill.

The adoption of goals by server-agents is formalised in the next schema, in which a server-agent is ascribed an additional set of goals. It describes the alternative where the cup is serving as a paper weight and is then subsequently given the goal of storing flowers. The schema checks that the adopting agent is a server-agent in the system and that the new goals are distinct from the existing goals.

$$
\begin{array}{|l}
\hline
ServerAgentAdoptGoals \\
a? : ServerAgent \\
gs? : \mathbb{P}\, Goal \\
\Delta World \\
\hline
a? \in serveragents \\
gs? \cap a?.goals = \{\} \\
neutralobjects' = neutralobjects \\
serveragents' = serveragents \setminus \{a?\} \cup \{EntityAdoptGoals\ (a?, gs?)\} \\
autonomousagents' = autonomousagents \\
\hline
\end{array}
$$

In some situations, a server-agent is released from some but not all of its agency obligations. Suppose, for example, that a window is open in A's office and that a cup is being used as a paperweight by A and a vase by B. If the window is subsequently closed, then the cup may be released from its agency obligations as a paperweight but still remain an agent because it is holding flowers. Formally, the operation schema representing this change of agency obligation is specified in the next schema. Notice that the goals that are removed from an agent in this operation must be a *proper* subset of its goals. The server-agent, $sa?$, is removed from the set of server-agents and replaced with the agent that results from removing the goals, $gs?$, from $a?$.

$$
\begin{array}{|l}
\hline
ServerAgentReleaseGoals \\
sa? : ServerAgent \\
gs? : \mathbb{P}\, Goal \\
\Delta MultiAgentSystem \\
\hline
sa? \in serveragents \\
gs? \subset sa?.goals \\
neutralobjects' = neutralobjects \\
serveragents' = serveragents \setminus \{sa?\} \cup \{EntityRemoveGoals\ (sa?, gs?)\} \\
autonomousagents' = autonomousagents \\
\hline
\end{array}
$$

6.3 Goal Adoption by Autonomous Agents

In the example above, the second possibility for goal adoption by server-agents involves Bill persuading Anne to first release the cup from its existing agency. The cup would then become a neutral-object and could be instantiated as required by Bill. In general, such persuasion or negotiation may be more difficult than the direct physical action required for goal adoption in non-autonomous entities. Autonomous agents are motivated and as such, only participate in an activity and assist others if it is to their motivational advantage to do so (that is, if there is some motivational benefit). They create their own agendas and for them, goal adoption is a *voluntary* process as opposed to an *obligatory* one for non-autonomous agents. In a similar example, Anne might ask Bill to assist in moving a table, but Bill may refuse.

Formally, the operation of an autonomous agent adopting the goals of another is specified in the following schema where the set of autonomous agents is updated to include the newly instantiated target autonomous agent. Note that this does not detail the persuasion involved, but simply the state change resulting from the goal adoption.

$$
\begin{array}{|l}
\hline
AutonomousAgentAdoptGoals \\
\hline
AssessGoals \\
aa? : AutonomousAgent \\
gs? : \mathbb{P}\, Goal \\
\Delta World \\
\hline
aa? \in autonomousagents \\
autonomousagents' = autonomousagents \setminus \{aa?\} \cup \\
\qquad\qquad \{EntityAdoptGoals\ (aa?, gs?)\} \\
agents' = agents \\
objects' = objects \\
\neg\ (\exists\, hs : \mathbb{P}\, Goal \mid hs \subseteq goalbase \land hs \neq gs? \bullet \\
\quad satisfygenerate\ hs > satisfygenerate\ gs?) \\
\hline
\end{array}
$$

In general, goals must be adopted through explicit autonomous agent initiative, as opposed to an ascription of goals for non-autonomous agents. However, in some contexts the ascription of goals to autonomous agents may be meaningful. Suppose, as a dramatic yet unlikely example, that Anne incapacitates Bill in some way and places him by the door to function as a draft excluder. In this situation, the autonomous agent, Bill could be *ascribed* the goal of keeping out the draft even though he has not explicitly adopted this goal. Such cases can be described by considering the autonomous agent as an agent in an obligatory relationship. In this thesis, however, we restrict *autonomous goal adoption* to the explicit and voluntary generation of goals that have been recognised in others. In our view, this is the only case in which *cooperation* takes place as opposed to mere *engagement* [16].

6.4 Autonomous Goal Destruction

For a number of reasons an autonomous agent may destroy adopted goals. For example, suppose Anne wishes to move a table and has persuaded Bill to help. If Anne subsequently destroys some important agency relationship of Bill's, it is possible that Bill

may then destroy the goal he has adopted from Anne of moving the table. As with goal adoption, for an autonomous agent to destroy goals, this must be considered the most motivationally beneficial course of action. This scenario is formalised below and is similar to the previous schema.

$$
\begin{array}{|l}
\hline
AutonomousAgentDestroysGoals \\
aa? : AutonomousAgent \\
gs? : \mathbb{P}\, Goal \\
\Delta MultiAgentSystem \\
AssessGoals \\
\hline
aa? \in autonomousagents \\
gs? \subseteq aa?.goals \\
autonomousagents' = (autonomousagents \setminus \{aa?\}) \cup \\
\qquad\qquad \{EntityRemoveGoals\ (aa?, gs?)\} \\
agents' = agents \land objects' = objects \\
\hline
\end{array}
$$

7 Discussion

Social behaviour by which individual agents interact and cooperate is an intricate and complex process which both structures and depends on the distribution of goals in a global system. If target agents are already involved in significant social interactions with others, it may not be possible to initiate new interactions with them. If such new interaction is possible, however, this will change the distribution of goals among agents in the system, potentially impacting on existing relationships, and certainly imposing structure to constrain new ones. In any of these situations, such social behaviour can only arise through the generation of goals by one agent, and the adoption of goals by another. The preceding sections have provided an operational specification of these processes so that the roles of goals and motivations are clarified and explicated.

Using the Z notation provides a way to formalise these mechanisms which first removes ambiguity from the analysis, and second enables an easy transition to be made to practical systems. Moreover, the specification of goal generation and adoption is one part of a much larger yet integrated theory of agent interaction which covers agent taxonomy [14], formal systems specification [15,4], agent relationships [16], and the construction of implemented agent systems [16].

Research on motivation is currently being pursued from a variety of perspectives including psychology and ethology, while computational research into motivation is also significant. The notion of motivation is not new. Simon, for example, takes motivation to be "that which controls attention at any given time," [25]. Sloman [26] has elaborated on Simon's work, showing how motivations are relevant to emotions and the development of a computational theory of mind. Others have used motivation and related notions in developing computational architectures for autonomous agents such as the *motives* of Norman and Long [22], and the *concerns* of Moffat and Frijda [20]. What is new about the current work is the role of motivation in defining autonomy and in enabling goal generation and adoption.

The agent hierarchy distinguishes clearly between objects, agents and autonomous agents in terms of goals and motivations. Such an analysis of the entities in the world not only provides appropriate structures so that different levels of functionality may be established, but also information as to how multiple entities or agents can cooperate to solve problems which could not be solved alone. By basing the distinctions on function and purpose, we do not arbitrarily differentiate between cups and robots, for example, especially when it is not useful to do so. Instead, our motivation and goal based analysis allows us to concentrate precisely on important aspects of multi-agent interaction and problem-solving. In that context, we have considered the roles of goal generation and adoption. We have specified how and why goals must be generated in some autonomous agents in response to motivations, grounding chains of goal adoption, and further, how goals are adopted by objects, agents and autonomous agents in this agent model.

References

1. C. Balkenius. The roots of motivation. In J. Meyer, H. L. Roitblat, and S. W. Wilson, editors, *From animals to animats 2, Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. MIT Press/Bradford Books, 1993.
2. L. P. Beaudoin and A. Sloman. A study of motive processing and attention. In *Prospects for Artificial Intelligence: Proceedings of AISB93*, pages 229–238, Birmingham, 1993.
3. M. d'Inverno and M. Luck. A formal view of social dependence networks. In C. Zhang and D. Lukose, editors, *Distributed Artificial Intelligence Architecture and Modelling: Proceedings of the First Australian Workshop on Distributed AI, LNAI 1087*, pages 115–129. Springer, 1996.
4. M. d'Inverno and M. Luck. Formalising the contract net as a goal directed system. In W. Van de Velde and J. W. Perram, editors, *Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World, LNAI 1038*, pages 72–85. Springer, 1996.
5. M. d'Inverno and M. Luck. Development and application of a formal agent framework. In M. G. Hinchey and L. Shaoying, editors, *Proceedings of the First IEEE International Conference on Formal Engineering Methods*, pages 222–231. IEEE Press, 1997.
6. M. d'Inverno and M. Luck. Making and breaking engagements: An operational analysis of agent relationships. In C. Zhang and D. Lukose, editors, *Multi-Agent Systems Methodologies and Applications: Proceedings of the Second Australian Workshop on Distributed AI, LNAI 1286*, pages 48–62. Springer, 1997.
7. M. d'Inverno and M. Luck. Engineering AgentSpeak(L): A formal computational model. *Journal of Logic and Computation*, 8(3):233–260, 1998.
8. M. d'Inverno, M. Luck, and M. Wooldridge. Cooperation structures. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 600–605, Nagoya, Japan, 1997.
9. T. Halliday. Motivation. In T. R. Halliday and P. J. B. Slater, editors, *Causes and Effects*. Blackwell Scientific, 1983.
10. J. R. P. Halperin. Machine motivation. In J. A. Meyer and S.W. Wilson, editors, *Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats*, pages 238–246. MIT Press/Bradford Books, 1991.
11. R. A. Hinde. *Ethology: Its nature and relations with other sciences*. Fontana Press, 1982.
12. Z. Kunda. The case for motivated reasoning. *Psychological Bulletin*, 108(3):480–498, 1990.
13. M. Luck. *Motivated Inductive Discovery*. PhD thesis, University of London, 1993.

14. M. Luck and M. d'Inverno. A formal framework for agency and autonomy. In *Proceedings of the First International Conference on Multi-Agent Systems*, pages 254–260. AAAI Press / MIT Press, 1995.
15. M. Luck and M. d'Inverno. Structuring a Z specification to provide a formal framework for autonomous agent systems. In J. P. Bowen and M. G. Hinchey, editors, *The Z Formal Specification Notation, 9th International Conference of Z Users, LNCS 967*, pages 48–62. Springer, 1995.
16. M. Luck and M. d'Inverno. Engagement and cooperation in motivated agent modelling. In *Distributed Artificial Intelligence Architecture and Modelling: Proceedings of the First Australian Workshop on Distributed AI, LNAI 1087*, pages 70–84. Springer, 1996.
17. P. Maes. The dynamics of action selection. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 991–997, Detroit, 1989.
18. P. Maes. How to do the right thing. *Connection Science*, 1(3):291–323, 1989.
19. P. Maes. A bottom-up mechanism for behaviour selection in an artificial creature. In J. A. Meyer and S.W. Wilson, editors, *Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats*, pages 238–246. MIT Press/Bradford Books, 1991.
20. D. Moffat and N. H. Frijda. Where there's a will there's an agent. In M. Wooldridge and N. R. Jennings, editors, *Intelligent Agents: Theories, Architectures, and Languages, LNAI 890*, pages 245–260. Springer, 1995.
21. D. Moffat, N. H. Frijda, and R. H. Phaf. Analysis of a model of emotions. In *Prospects for Artificial Intelligence: Proceedings of AISB93*, pages 219–228, Birmingham, 1993.
22. T. J. Norman and D. Long. Goal creation in motivated agents. In M. Wooldridge and N. R. Jennings, editors, *Intelligent Agents: Theories, Architectures, and Languages, LNAI 890*, pages 277–290. Springer, 1995.
23. T. J. Norman and D. Long. Alarms: An implementation of motivated agency. In M. Wooldridge, J.P. Muller, and M. Tambe, editors, *Intelligent Agents: Theories, Architectures, and Languages, LNAI 1037*, pages 219–234. Springer, 1996.
24. U. Schnepf. Robot ethology: A proposal for the research into intelligent autonomous systems. In J. A. Meyer and S.W. Wilson, editors, *Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats*, pages 465–474. MIT Press/Bradford Books, 1991.
25. H. A. Simon. Motivational and emotional controls of cognition. In *Models of Thought*, pages 29–38. Yale University Press, 1979.
26. A. Sloman. Motives, mechanisms, and emotions. *Cognition and Emotion*, 1(3):217–233, 1987.
27. A. Sloman and M. Croucher. Why robots will have emotions. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 197–202, Vancouver, B.C., 1981.
28. J. M. Spivey. *The Z Notation: A Reference Manual*. Prentice Hall, 2nd edition, 1992.
29. L. Steels. When are robots intelligent autonomous agents? *Journal of Robotics and Autonomous Systems*, 15:3–9, 1995.
30. D. L. Waltz. Eight principles for building an intelligent robot. In J. A. Meyer and S.W. Wilson, editors, *Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats*. MIT Press/Bradford Books, 1991.